The Dissertation Committee for Murat Kocaoglu
certifies that this is the approved version of the following dissertation:

Causality: From Learning to Generative Models

Committee:

Alexandros G. Dimakis, Co-supervisor

Sriram Vishwanath, Co-supervisor

Sanjay Shakkottai

Constantine Caramanis

Elias Bareinboim

Causality: From Learning to Generative Models

by

Murat Kocaoglu,

DISSERTATION

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY

THE UNIVERSITY OF TEXAS AT AUSTIN

August 2018

Dedicated to my mother Lütfiye and my father Cengiz.

Acknowledgments

The cause of an outcome is seldom a single variable. This thesis would not be possible had either of the following contributors not been there.

My parents, Lütfiye Ay Kocaoğlu and Cengiz Kocaoğlu have always been there for me, helped me make the right decisions, have been and will always be the main actors in everything I do. While I grew up, they have been the greatest role models I could have asked for. They presented me with plenty of opportunities and protected me at all costs. Their unconditional love and support have always been the stable harbor that I could turn to. My mother Lütfiye have taught me the importance of communication, social skills and understanding the others. My father Cengiz, have taught me the importance of grace, tranquility and blessed me with the love for science.

I am very lucky to have Alex Dimakis and Sriram Vishwanath as my PhD advisors. Alex, with his endless enthusiasm for research and realism have been one of the most influential role models. He has always been truthful, shown constant support and patience and equipped me with the most important skills for research: how to always ask the right questions and look for answers that can make a difference. Sriram, with his faith and trust have always encouraged me to chase the path I choose in research and in life. My meetings with him have been uplifting and equipped me with the motivation I needed to keep

trying. I cannot thank both enough for having always been there for me with their support and for all the life lessons that they have taught me which I would not have learnt otherwise.

I am very grateful for the prolific, collaborative and the extremely friendly environment of UT Austin and specifically WNCG. The people I encountered here made me enjoy every second I spent in PhD. I would like to thank my collaborators. A special thanks goes to Karthikeyan Shanmugam: He was my first student collaborator in PhD and I have learned so much working with him on multiple projects. He has always amazed me with his brilliance and passion to this day. My collaboration with Babak Hassibi, Chris Snyder, Elias Bareinboim, Sanjay Shakkottai and Constantine Caramanis have been instrumental in making this thesis possible. I would like to also express my gratitude to my committee members Sanjay Shakkottai, Constantine Caramanis and Elias Bareinboim. Their guidance has been an important element in how this thesis took its final form.

I was lucky to have made lifelong friendships during my studies: A special thanks goes to Ankit S. Rawat who has been there for me when I needed the most and introduced me to the realm of working in coffee shops. I cherish the long debates we had with Avradeep Bhowmik, Virag Shah and Srinadh Bhojanapalli, and the chess nights we shared with Karthikeyan Shanmugam, Mandar Kulkarni and Chris Snyder. I enjoyed the debates we had comparing our cuisines with Sara Mourad and Eirini Asteri, spiced with a pinch of joyful teasing of one another. I appreciate the laughs we had with Erik Lindgren,

who is one of the most uniquely beautiful people I know. I enjoyed the long philosophical and technical debates we had with Abishek Sankararaman. Rajat Sen and Mridula Maddukuri have been great friends with whom I shared many great moments inside and outside the lab. I am grateful to have met Ajil Jalal, who has been an amazing friend, has been there for me and supportive and understanding. The table tennis matches we had with Natasa Dragovic have been a great source of fun. I am very lucky to have responded to the mail Koray Uğur Erbaş had Melanie send out, which established our lifelong friendship. Vatsal Shah has been a great companion to talk to in our late-night coffee shop sessions. The talks we had with Rajiv Khanna, small or deep, have been greatly influential. Ethan Elenberg has influenced me with his hard-working nature and his mastery in work-life balance. I appreciate the times I run into Soumya Basu, always a friendly face, in Coffee Medici on West Lynn. I enjoyed the chats we had with Taewan Kim during our breaks. I appreciate the fun times we shared with Dave Van Veen, Ashish Katiyar, Justin Lewis, Ahmad Alammouri, Mónica Ribero and Jessica Hoffman. I am lucky to have become close friends with Shalmali Joshi in my final few months in PhD, one that came late, but one to last. From her, I learned about many different aspects of life, and a variety of new ways to enjoy it. Her endless support and strong character have helped me deal with problems with ease.

I would like to thank Melanie Gulick, who has one of the most colorful personalities that I know, Melody Singleton, Karen Little and Apipol Piman for always helping me out with the support I needed to handle the administrative

work.

I also would like to express my gratitude to my M.S. advisor Özgür Barış Akan. He has exposed me to research for the very first time and have been very supportive of my decision to pursue a PhD in the US. This journey would not have started without his support. I would also like to thank the Scientific and Technological Research Council of Turkey (TÜBİTAK) for their support during my studies in Turkey.

Causality: From Learning to Generative Models

Publication No. _____

Murat Kocaoglu, Ph.D.
The University of Texas at Austin, 2018

Supervisors: Alexandros G. Dimakis
Sriram Vishwanath

Causality is a fundamental concept in multiple disciplines. Causal questions arise in fields ranging from medical research to engineering, philosophy to physics. The last few decades have witnessed the development of a mathematical model of probabilistic causation by Judea Pearl and many others. In this modeling framework, directed acyclic graphs arise as natural objects to capture causal relations between random variables. The directed acyclic graph that captures the causal relations between variables is called the *causal graph* of the system.

A fundamental problem is *to learn* the causal graph over a set of observed variables. In this thesis, we propose new algorithms for learning causal graphs in various settings: *1)* First, we consider the setting where the observational data is available, but we are not allowed to perform new experiments without any unobserved common causes in Chapter 2, and with an unobserved common cause in Chapter 3 on two discrete/categorical variables. *2)* Second, we

consider the scenario where we are allowed to perform experiments, but these experiments have a cost associated with them and our goal is to minimize this cost for learning the causal graph in Chapter 4, and when there are unobserved common causes and we want to minimize the number of experiments to learn both the causal graph on the observed variables and the location of latents in Chapter 5. After the causal graph is learned, the next problem is to fit a functional model that can sample from the causal model. *3)* Third, in Chapter 6 we suggest the use of neural networks, specifically generative adversarial networks for learning the causal model when the causal graph is known without latent variables.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Causality is one of the fundamental concepts in science and philosophy. It is essential for understanding the natural mechanism and planning. There are various frameworks for modeling causal systems. One of the two well established frameworks is the potential outcomes framework developed by Rubin [59], which is frequently used in the statistics literature today. The main objective is to deal with the phenomenon called *the fundamental problem of causal inference*: A patient can only be assigned a single treatment, and we never get to see the outcome of a different assignment for that particular patient. Various techniques are used to estimate the average causal effect of a treatment within the Rubin model (also called Neyman-Rubin model). The second framework is developed by Judea Pearl and others. The key ingredient of the Pearlian framework is the use of directed graphs for representing the causal relations between a set of random variables, where each node of the graph is a random variable. Pearl developed a mathematical theory of causation, which can model the effect of randomized experiments. His theory extends Bayesian networks by equipping them with the capability to capture causal relations.

In this thesis, we operate within the Pearlian regime. We are interested in developing novel techniques for learning the causal relations between variables, which boils down to learning the causal graph.

The causal inference techniques for learning causal graphs can be divided into two large categories[1]: Data-driven learning techniques and experimental (interventional) learning techniques.

Data-driven techniques attempt to extract the most information about the causal structure given observations (samples) of the variables in a system. This thread may have started from the independent discovery of the IC [135] and PC algorithms [126], which almost identically, and contrary to previously held beliefs, showed the possibility of recovering causal relations from purely observational, non-experimental data. These algorithms are also called *constraint-based* as they find the class of equivalent causal graphs that satisfy the conditional independence (CI) statements that hold in the data. Although these methods work on general causal graphs and helps us extract the most complete causal picture one can obtain from observational data using CI tests, additional assumptions on the data generating model may help us identify more than what constraint-based methods can offer. For example, if we assume that the data comes from an Additive Noise Model (ANM), i.e., if the exogenous variables' influence on the effect variable is additive, causal direction between two variables can be identified except for a measure zero set

[1]A third thread contains the score-based techniques that identify the DAG that maximizes a score such as likelihood. In this thesis, we are not interested in this regime.

of parameters [54], whereas IC/PC cannot be used in this case. Along these lines a plethora of methods are developed [124, 54, 81, 106].

Experimental learning techniques can be used when PC/IC algorithms are insufficient [2], or if we cannot make extra assumptions such as additive noise models, and we can set-up, perform experiments and collect additional data. As these experiments are costly, one important research problem is to design the set of experiments that allows us to reason about the underlying causal system with minimum number of interventions [34].

We contribute to both literatures by proposing new algorithms to infer causal relations under various setups. Our first two contributions are on learning the causal graph between two discrete variables with and without causal sufficiency using *entropy*. The next contribution is on designing experiments for learning the underlying causal graph that optimize the total experimental cost. The next contribution is on designing an efficient set of interventions to learn the causal graph when latent variables are present in the system. Finally we design an adversarial training scheme for learning causal implicit generative models, when the causal graph is known. We specifically focus on the application of image generation from labels, where we see this as a causal process and develop architectures to sample from both observational and interventional image distributions.

[2]This can for example happen when the causal graph is a tournament, i.e., when there is no non-edge in then causal graph.

1.1 Learning from Data: Entropic Causal Inference

Our first contribution is within the observational learning framework. As mentioned before, within the Pearlian framework [105] it is possible to discover some causal directions between variables using only observational data with conditional independence tests. The PC algorithm [126] and its variants fully characterize which causal directions can be learned in the general case. For large graphs, GES algorithm [23] provides a score-based test to greedily identify the highest scoring causal graph given the data. Unfortunately, these approaches do not guarantee the recovery of true causal direction between every pair of variables, since typically data could be generated by several statistically equivalent causal graphs.

A general solution to the causal inference problem is to conduct experiments, also called interventions. An intervention forces the value of a variable without affecting the other system variables. This removes the effect of its causes, effectively creating a new causal graph. These changes in the causal graph create a post-interventional distribution among variables, which can be used to identify some additional causal relations in the original graph. The procedure can be applied repeatedly to fully identify any causal graph [49], [50], [56], [122].

Unfortunately, for many problems, it can be very difficult to create interventions since they require additional experiments after the original data collection. Researchers would still like to discover causal relations between variables using only observational data, using so-called data-driven causality.

Several recent works [22, 120] have developed such methods. To be able to make any conclusions on causal directions in this case, additional assumptions must be made about the mechanisms that generate the data.

We consider the problem of identifying the causal direction between two discrete random variables using observational data. Unlike many of the previous works, we keep the most general functional model but make an assumption on the unobserved exogenous variable: Inspired by Occam's razor, we assume that the exogenous variable is *simple* in the true causal direction. We quantify simplicity using Rényi entropy. Our main result is that, under natural assumptions, even though the exogenous variable has low H_0 entropy (cardinality) in the true direction, it must have high H_0 entropy in the wrong direction. We establish several algorithmic hardness results about estimating the minimum entropy exogenous variable. We show that the problem of finding the exogenous variable with minimum entropy is equivalent to the problem of finding minimum joint entropy given n marginal distributions, also known as the minimum entropy coupling problem. We propose an efficient greedy algorithm for the minimum entropy coupling problem that provably finds a local optimum. This gives a greedy algorithm for finding the exogenous variable with minimum H_1 (Shannon Entropy). Our greedy entropy-based causal inference algorithm has similar performance to the state of the art additive noise models in real datasets. One advantage of our approach is that we make no use of the values of the random variables but only their distributions. Our method can therefore be used for causal inference for both ordinal and categorical data,

unlike additive noise models.

1.2 Learning from Data: Entropic Latent Variable Discovery

One of the most important challenges for the current causal inference algorithms is the existence of latent (unobserved) variables. Latent variables are especially prevalent in biology: Biological signals, such as blood pressure, or fMRI images, are confounded by unobserved factors, see e.g. [46] and references therein.

Consider the following problem: we are given two discrete random variables X, Y over m and n states respectively. Suppose we want to construct a third random variable Z, such that X and Y are independent conditioned on Z. Without any constraints this can be trivially achieved: Simply picking $Z = X$ or $Z = Y$ ensures that $X \perp\!\!\!\perp Y \,|Z$. However, this requires that the random variable Z is as *complex* as X or Y. We therefore ask the following question: *is there a simple Z that makes X, Y conditionally independent?* Suppose we measure the complexity of Z by its cardinality (equivalently, its Renyi entropy for $\alpha = 0$). Then, a non-trivial answer to this question would require us to find a random variable Z with k states, where $k < \min\{m, n\}$ that renders X, Y conditionally independent. This problem of recovering a small cardinality latent variable Z is closely related to Probabilistic Latent Semantic Analysis (pLSA), non-negative matrix factorization (NMF) and Latent Dirichlet Allocation (LDA). In this chapter we show that a solution to this problem can

also be used for causal inference in the presence of latent variables: Suppose, given two dependent random variables X, Y, that our objective is to decide if there is a direct causal relation between the two, or if the observed correlation is spurious, i.e., due to an indirect path via a latent variable.

Our main theoretical result is that it is always possible to distinguish the latent causal graph from the triangle causal graph (see Figure 3.1) where Z is latent, if the latent variable has *cardinality (aka Renyi entropy for $\alpha = 0$)* less than $\min\{m, n\}$, except for a measure zero set of joint distributions. We also study this problem using an alternate simplicity metric of *Shannon entropy* instead of cardinality. In this case, our problem becomes: Given two discrete variables X, Y with m, n states, respectively, what is the minimum entropy variable Z with k states that assures $X \perp\!\!\!\perp Y \,|Z$? We conjecture that an identifiability result also holds in this case even when $k \geq n$, thus permitting us to study a larger class of latent Z. More precisely, we expect most distributions generated from the triangle graph with small entropy Z to require a large entropy Z to satisfy conditional independence $X \perp\!\!\!\perp Y \,|Z$.

With these observations, our main algorithmic contribution is as follows: To solve the problem of recovering the minimum entropy Z that renders the observed variables X, Y conditionally independent, we propose an algorithm to minimize the loss $I(X; Y|Z) + \beta H(Z)$. The parameter β allows us to discover a tradeoff between *the simplicity of variable Z* and *how much of the dependence between X, Y it can explain away.* We show that our algorithm always outputs a stationary point of the loss function. Moreover, for $\beta = 1$, we are able to show

convergence of the algorithm, and that it converges to either a local minimum or a saddle point. We also empirically demonstrate that it converges much faster than gradient descent and recovers better solutions after convergence.

1.3 Learning with Experiments: Cost-Optimal Learning of Causal Graphs

Randomized trials are the golden standard for causal inference since they lead to reliable conclusions with minimal assumptions. The problem is that enforcing randomization to different variables in a causal inference problem can have significant and varying costs. A causal discovery algorithm should take these costs into account and optimize experiments accordingly.

In this chapter, we formulate this problem of learning a causal graph when there is a cost for intervening on each variable. We use interventions, i.e., experiments as described in the structural equation modeling framework [105, 126]. To perform each intervention, a scientist randomizes a set of variables and collects new data from the perturbed system. For example, suppose the scientist wants to discover the causal graph between a set of patient features, such as diet and blood sugar, and diabetes. Suppose she decides to perform an intervention on the *diet* variable. This entails forcing the desired dietary restrictions on a random subset of the participating patients. Next, suppose she decides to perform an intervention on the *blood sugar* variable. This intervention requires the scientist to adjust the blood sugar directly, for example through injection of glucose rather than through diet control. An intervention on the blood sugar is

arguably harder to perform than a dietary restriction. Hence, the blood sugar variable should be assigned a larger intervention cost than the diet variable. Performing an intervention on the variable *diabetes* is impractical and also unethical. Hence it should be potentially given the cost of infinity.

For learning the causal relations, randomized experiments can be used in the most general setting, where we do not have to make assumptions on the data generating model[3]. Recently, researchers have been interested in developing efficient experimental designs, i.e., the set of experiments to perform on the physical system for learning the causal graph associated with the system most efficiently [56, 50]. The problem of finding the minimum experimental design for learning a causal graph is motivated by the fact that experiments are in general costly: Each experiment requires a new population of units and administration of the treatment on this population. Researchers have also been interested in finding the minimum intervention design under the constraint that each intervention randomizes a small number of variables [122]. However, a general framework that can incorporate various types of interventional costs have not been proposed.

In the second part of the chapter, we consider the problem of learning a causal graph over a set of variables with interventions, where each variable is associated with a cost. We study the cost-optimal causal graph learning problem: For a given skeleton (undirected version of the causal graph), design

[3]Clearly, Pearlian framework is associated with a set of assumptions, e.g., the causal graph is acyclic.

the set of interventions with minimum total cost, that can uniquely identify any causal graph with the given skeleton. We show that this problem is solvable in polynomial time. Later, we consider the case when the number of interventions is limited. For this case, we provide polynomial time algorithms when the skeleton is a tree or a clique tree. For a general chordal skeleton, we develop an efficient greedy algorithm, which can be improved when the causal graph skeleton is an interval graph.

1.4 Learning with Experiments: Experimental Design for Learning Causal Graphs with Latent Variables

One of the most popular assumptions in causal inference research is that the data-generating model is *causally sufficient*, which means that no latent (unmeasured) variable affecting more than one observed variable exists. In practice, this is a very stringent condition since the existence of latents affecting more than one observed variable, and generating what is called *confounding bias*, is one of the main concerns of empirical scientists that do not believe that association is the same as causation. The problem of causation is deemed challenging in most of the empirical fields because scientists recognize that not all the variables influencing the observed phenomenon being studied can be measured. The general question that arises is then how much of the surface, observed behavior of the system is truly causal, or whether it is due to some external, unobserved forces (Pearl, 2000). Answering this question in a principled way has far-reaching implications for exploring and understanding

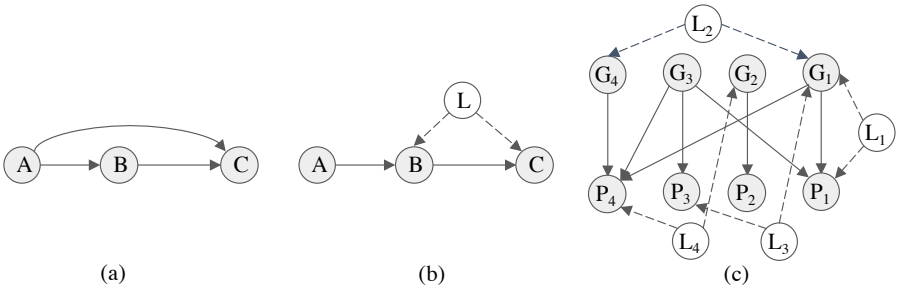


Figure 1.1: (a), (b): Different causal graphs that cannot be distinguished using observational data. (c): A bipartite causal graph between genes and phenotypes with multiple latent variables.

how nature works, and for better decision-making.

To account for the latent variables, the IC* [135] and FCI [126] algorithms were introduced, which showed the possibility of recovering causal structures *even when* latent variables may be confounding the observed behavior [4]. One of the main challenges faced by this family of algorithms is that only ancestral relations can be learned – namely, whether intervening on a variable has a causal influence on another, but not necessarily direct. For instance, Fig. 1.1(a),(b) depict two data-generating models that is not distinguishable from passive data whether A directly causes C or only indirectly (there exists no separator between A and C due to the presence of the latent L). Despite the practical challenges of recovering ancestral relations (e.g., finite samples, selection bias, missing data), we now have a complete characterization of what structures are recoverable from observational data [2, 126, 143].

[4]Hereafter, we refer to a *latent variable* as any variable that is not measured and affect more than one observed variable. In Fig. 1.1(b), note the latent variable L confounding the relationship between B and C.

In fact, inferences will be constrained within some type of equivalence class, i.e., a collection of models that are all compatible with the constraints implied over the observational distribution. Some works attempted to use interventional data to move from the equivalence class to a specific graph, but almost exclusively considering causally sufficient systems [34, 49, 50, 122, 132, 107]. A number of assumptions trying to identify plausible causal structures given interventional data collected a priori were also studied [125, 118, 107, 103], but without the goal of designing optimal interventions.

In this chapter, we generalize these results and propose the first algorithm to learning a causal graph with latent variables using interventions in a non-parametric regime. It is known that $\log(n)$ interventions are necessary and sufficient to learn a causal graph without latent variables [50], and we show, perhaps surprisingly, that there exists an algorithm that can learn a causal graph that is not very far from this result. More specifically, our contributions are as follow:

- We introduce a deterministic algorithm that can learn any causal graph and the existence and location of the latent variables using $\mathcal{O}(d\log(n) + l)$ interventions, where d is the largest degree and l is the longest directed path of the causal graph.
- We then design a randomized algorithm that can learn the observable graph and all the latent variables using $\mathcal{O}(d\log^2(n) + d^2\log(n))$ interventions with high probability, where d is the largest degree.

The first algorithm should be useful in practical settings where the longest

directed path is not very deep (smaller than $\log(n)$), which include bipartite, time-series, and relational type of domains where the number of variables is potentially high but the topology is somewhat sparse. As an example application, consider the problem of inferring the causal effect of a set of genes on a set of phenotypes, that could be cast as learning a bipartite causal system (Fig. 1.1(c)). For settings where the data-generating model is dense, we introduce a randomized algorithm that with high probability is capable of unveiling the true causal structure.

1.5 Learning Causal Implicit Generative Models

The techniques described so far are interested in learning the causal graph between variables. However, these cannot be used to learn the causal model, i.e., the functional relations between variables. Although this problem is not identifiable in general, it is of interest to construct a causal generative model which would allow us to sample from interventional distributions.

Generative adversarial networks are neural generative models that can be trained using backpropagation to mimick sampling from very high dimensional nonparametric distributions [42]. A *generator* network models the sampling process through feedforward computation. The generator output is constrained and refined through the feedback by a competitive "adversary network", that attempts to discriminate between the generated and real samples. In the application of sampling from a distribution over images, a generator, typically a neural network, outputs an image given independent noise variables. The

objective of the generator is to maximize the loss of the discriminator (convince the discriminator that it outputs images from the real data distribution). GANs have shown tremendous success in generating samples from distributions such as image and video [136] and have even been proposed for language translation [140].

One extension idea for GANs is to enable sampling from the class conditional data distributions by feeding labels to the generator. Various neural network architectures have been proposed for solving this problem [94, 101, 5]. As far as we are aware of, in all of these works, the class labels are chosen independently from one another. Therefore, choosing one label does not affect the distribution of the other labels. As a result, these architectures do not provide the functionality to condition on a label, and sample other labels and the image. For concreteness consider a generator trained to output images of birds when given the *color* and *species* labels. On one hand, if we feed the generator *color=blue*, since *species* label is independent from the *color* label, we are likely to see blue eagles as well as blue jays. However, we do not expect to see any blue eagles when conditioned on *color=blue* in any dataset of real bird images. Similarly, consider a generator trained to output face images given the *gender* and *mustache* labels. When labels are chosen independently from one another, images generated under *mustache = 1* should contain both males and females, which is clearly different than conditioning on *mustache = 1*. The key for understanding and unifying these two notions, conditioning and being able to sample from distributions different than the dataset's is to use *causality*.

We can think of generating an image conditioned on labels as a causal process: Labels determine the image distribution. The generator is a functional map from labels to image distributions. This is consistent with a simple causal graph *"Labels cause the Image"*, represented with the graph $L \rightarrow G$, where L is the set of labels and G is the generated image. Using a finer model, we can also include the causal graph between the labels. Using the notion of causal graphs, we are interested in extending the previous work on conditional image generation by *(a) capturing the dependence* and *(b) capturing the causal effect* between labels and the image.

As an example, consider the causal graph between gender (G) and mustache (M) labels. The causal relation is clearly *gender causes mustache*[5], shown with the graph $G \rightarrow M$. Conditioning on *gender=male*, we expect to see males with or without mustaches, based on the fraction of males with mustaches in the population. When we condition on *mustache = 1*, we expect to sample from males only since the population does not contain females with mustaches. In addition to sampling from conditional distributions, causal models allow us to sample from various different distributions called *interventional distributions*, which we explain next.

From a causal lens, using independent labels corresponds to using an empty causal graph between the labels. However in practice the labels are

[5]In reality, there may be confounder variables, i.e., variables that affect both, which are not observable. In this chapter, we ignore this effect by assuming the graph has causal sufficiency, i.e., there does not exist unobserved variables that cause more than one observable variable.

not independent and even have clear causal connections (e.g., *gender* causes *mustache*). Using an empty causal graph instead of the true causal graph, and setting a label to a particular value is equivalent to *intervening* on that label in the original causal graph, but also ignoring the way it affects other variables. An intervention is an experiment which fixes the value of a variable, without affecting the rest of the causal mechanism, which is different from conditioning. An intervention on a variable affects its descendant variables in the causal graph. But unlike conditioning, it does not affect the distribution of its ancestors. For example, instead of the causal graph *Gender causes Mustache*, if we used the empty causal graph between the same labels, intervening on *Gender* = *Female* would create females with mustaches, whereas with the correct causal graph, it should only yield females without mustaches since setting the *Gender* variable will affect all the variables that are downstream, e.g., *mustache*. Similarly, for generating birds with the causal graph *Species causes color*, intervening on *color* = *blue* allows us to sample blue eagles (which do not exist) whereas conditioning on *color* = *blue* does not.

An *implicit* generative model [95] is a mechanism that can sample from a probability distribution but cannot provide likelihoods for data points. In this chapter we propose *causal implicit* generative models (CiGM): mechanisms that can sample not only from probability distributions but also from *conditional* and *interventional* distributions. We show that when the generator structure inherits its neural connections from the causal graph, GANs can be used to train causal implicit generative models. We use WassersteinGAN to train a causal

implicit generative model for image labels, as part of a two-step procedure for training a causal implicit generative model for the images and image labels. For the second step, we propose a novel conditional GAN architecture and loss function for adversarial training. We show that the global optimal generator can sample from the correct conditional and interventional distributions, which is shown by Theorem 17. We also show that the trained causal implicit generative model for the labels concatenated with our conditional GAN is a causal implicit generative model for the labels and the image (Corollary 5).

Chapter 2

Entropic Causality

The most popular assumption for two-variable data-driven causality is the additive noise model (ANM) [124]. In ANM, any outside factor is assumed to affect the effect variable additively, which leads to the equation $Y = f(X) + E, E \perp\!\!\!\perp X$. Although restrictive, this assumption leads to strong theoretical guarantees in terms of identifiability, and provides the state of the art accuracy in real datasets. [124] showed that if f is linear and the noise is non-Gaussian the causal direction is identifiable. [54] showed that when f is non-linear, irrespective of the noise, identifiability holds in a non-adverserial setting of system parameters. [108] extended ANM to discrete variables.

Another approach is to exploit the postulate that the cause and mechanism are in general independently assigned by nature. The notion of *independence* here is vague and one needs to assign maps, or conditional distributions to random variables to argue about independence of cause and mechanism.

This chapter is based on the material from the publications [69, 70]: M. Kocaoglu, Alex Dimakis, Sriram Vishwanath, Babak Hassibi *"Entropic Causal Inference,"* Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17), 2017 and M. Kocaoglu, Alex Dimakis, Sriram Vishwanath, Babak Hassibi *"Entropic Causality and Greedy Minimum Entropy Coupling,"* IEEE International Symposium on Information Theory (ISIT), pp. 1465-1469, Aachen, Germany, 2017. The author of this dissertation contributed to the conception of the research problem, theoretical developments and experimental validation.

In this direction an information-geometry based approach is suggested [61]. Independence of cause and mechanism is captured by treating the log-slope of the function as a random variable, and assuming that it is independent from the cause. In the case of a deterministic relation $Y = f(X)$, there are theoretical guarantees on identifiability. However, this assumption is restrictive for real data.

Previous work exploited these two ideas, additive noise, and independence of cause and mechanism, to draw data-driven causal conclusions about problems in a diverse range of areas from astronomy to neuroscience [120], [118]. [120] uses the same idea that the cause and effect are independent in the time series of a linear filter. They suggest the spectral independence criterion, which is robust to time shifts. [22] uses kernel space embeddings with the assumption that the cause distribution $p(x)$ and mechanism $p(y|x)$ are selected independently to distinguish cause from effect.

As noted by [22], although conceptually proposed before, using Kolmogorov complexity of the factorization of the joint distribution $p(y|x)p(x)$ and $p(x|y)p(y)$ as a criterion for deciding causal direction has not been used successfully until now.

The use of information theory as a tool for causal discovery is currently gaining increasing attention. This is through different appoaches, e.g., for time-series data, Granger causality and Directed Information can be used [45, 37, 110], see also [89]. However, researchers have not used entropy as a measure of simplicity in the causal discovery literature, probably because the

entropies $H(Y|X)$ and $H(X|Y)$ do not give us any more information than $H(X)$ and $H(Y)$, due to the symmetry $H(Y) + H(X|Y) = H(X) + H(Y|X)$. In our work, as we will explain, we minimize $H(E)$ which initially sounds similar, *but is fundamentally different* from $H(Y|X)$. Entropy has found some additional uses in the causality literature recently: In [40], authors use maximum mutual information between X, Y in order to quantify the causal strength of a known causal graph.

The work that is most similar to ours in spirit is [97], which also drops the additive noise assumption. Their approach and setup are different in many ways: Authors work with continuous data. To be able to handle this generic form, they have to make strong assumptions on the exogenous variable, function, and distribution of the cause: [97] assume that the exogenous variable is a standard Gaussian, a Gaussian mixture prior for the cause, and a Gaussian process as the prior of the function.

2.1 Our contributions

In this chapter, we propose a novel approach to the causal identifiability problem for discrete variables. Similar to [97], we keep the most general functional model, but only put an assumption on the exogenous (background) variable. Based on Occam's razor, we employ a simplicity assumption on the unobserved exogenous variable. We use Rényi entropy, which is defined as $H_a(X) = \frac{1}{1-a} \log \left(\sum_i p_i^a\right)$, for a random variable X with state probabilities p_i. We focus on two special cases of Rényi entropy: H_0, which corresponds to

the logarithm of the number of states, and H_1 which corresponds to Shannon entropy, but our framework can be extended.

Specifically, if the true causal direction is $X \to Y$, then the random variable Y is an arbitrary function of X and an exogenous variable E: $Y = f(X, E)$ where E is independent from the cause X. Our key assumption is that the exogenous variable E is *simple*, i.e., has low Rényi entropy. The postulate is that for any model in the wrong direction $X = f'(Y, \tilde{E})$, the exogenous variable $\tilde{E}$ has high Rényi entropy. We are able to prove this result for the H_0 special case of Rényi entropy, assuming generic distributions for X, Y. Furthermore, we empirically show that using H_1 Shannon entropy we obtain practical causality tests that work with high probability in synthetic datasets and that slightly outperforms the previous state of the art in real datasets.

Our assumption is an entropic interpretation of Occam's razor, motivated by what E represents in the causal model. The exogenous variable captures the combined effect of all the variables not included in the system model, which affect the distribution of Y. Our causal assumption can be stated as *"there should not be too much complexity not included in the causal model"*. For $a \to 1$, *i.e.*, Shannon entropy, $H(X) + H(E)$, $H(Y) + H(\tilde{E})$ are the number of random bits required to generate an input for the causal system $X \to Y$ and $X \leftarrow Y$, respectively. The simplest explanation of an observed joint distribution, i.e., the direction which requires nature to generate smaller number of random bits is selected as the true causal model. More precisely we have the following:

Assumption 1. *Entropy of the exogenous variable E is small in the true causal direction.*

The notions of simplicity that we consider are H_0, which is log-cardinality, and H_1, which is Shannon entropy. One significant advantage of using Shannon entropy as a simplicity metric is that it can be estimated more robustly in the presence of measurement errors, unlike cardinality H_0.

We prove an identifiability result for H_0 entropy, i.e., cardinality of E: If the probability values are not adversarially chosen, for most functions, the true causal direction is identifiable under Assumption 1. Based on experimental evidence, we conjecture that a similar identifiability result must hold for Shannon entropy H_1.

To use our framework we need algorithms that explain a dataset by finding an exogenous variable E with minimum cardinality H_0 and minimum Shannon entropy H_1. Since the entropies of X and Y can be very different, any metric to determine the true causal direction cannot only consider the entropy of the exogenous variable without incorporating the entropy of the cause. We explain the exogenous variable in both directions and declare the causal direction to be the one with the smallest joint entropy $H_a(X) + H_a(E)$ versus $H_a(Y) + H_a(\tilde{E})$. Our method can be applied for any Rényi entropy H_a but in this chapter we only use $a = 0$ and $a = 1$.

Unfortunately, minimizing $H_0(E)$ seems very hard for real datasets since it offers no noise robustness. For Shannon entropy we can do much better for

real data. The first step in obtaining a practical algorithm is showing that the minimum H_1 explanation is equivalent to the following problem: For n random variables with given marginal distributions, find a joint distribution with the minimum Shannon entropy that is consistent with the given marginals. This problem is called the minimum Shannon entropy coupling and is known to be NP hard [75]. We propose a greedy approximation algorithm for this problem that empirically performs very well. We also prove that, our algorithm always produces a local minimum.

In summary our contributions in this chapter include:

- We show identifiability for generic low-entropy causal models under Assumption 1 with H_0.
- We show that the problems of identifying the minimum cardinality (H_0) exogenous variable, and identifying the minimum Shannon entropy (H_1) exogenous variable given a joint distribution are both NP hard.
- We design a novel greedy algorithm for the minimum entropy coupling problem, which turns out to be equivalent to the problem of finding exogenous variable with minimum H_1 entropy.
- We empirically validate the conjecture that the causal direction is identifiable under Assumption 1 with H_1, using experiments on synthetic datasets.
- We empirically show that our causal inference algorithm based on Shannon entropy minimization has slightly better performance than the existing best algorithms on a real causal dataset. Interestingly, our algorithm

uses only the probability distributions rather than the actual values of the random variables, and hence is applicable to categorical variables.

2.2 Background and Notation

A tuple $\mathcal{M} = (X, U, \mathcal{F}, D, p)$ is a causal model when, 1) $\mathcal{F} = \{f_i\}$ are deterministic functions, 2) $X = \{X_i\}$ are a set of endogenous (observed) variables $U = \{U_i\}$ are a set of exogenous (latent) variables with $X_i = f_i(Pa_i, U_i), \forall i$ where Pa_i are the endogenous parents and U_i is the exogenous parent of X_i in directed acyclic graph D, 3) U are mutually independent with respect to p. The observable variable set X has a joint distribution implied by the distributions of U, and the functional relations f_i. D is then a Bayesian network for the induced joint distribution of endogenous variables. A standard assumption employed in Pearl's model *causal sufficiency* is also used here: Every exogenous variable is a direct parent of at most one endogenous variable.

In this chapter, we consider a simple two variable causal system which contains only two endogenous variables X, Y. Assume X causes Y, which is represented as $X \to Y$. The model is determined only by one exogenous variable E, and a function f, where $Y = f(X, E)$. The probability distribution of X and E, and f determines the distribution of Y. This model is shown by the tuple $\mathcal{M} = (\{X, Y\}, E, f, X \to Y, p_{X,E})$. Notice that we do not assign an exogenous variable to X, since it is the source node in the graph.

We denote the set $\{1, 2, \cdots, n\}$ by $[n]$. $\sum_i x_i$ is meant to run through every possible index. log refers to the logarithm base 2. For two variables X, Y,

$\mathbf{Y}|\mathbf{X}$ and $\mathbf{X}|\mathbf{Y}$ denote the conditional probability distribution matrices, i.e., $\mathbf{Y}|\mathbf{X}(i,j) = p(y = i|x = j)$ and $\mathbf{X}|\mathbf{Y}(i,j) = p(x = i|y = j)$. The statistical independence of two random variables X and E are shown by $X \perp\!\!\!\perp E$. For notational convenience, probability distribution of random variable X is shown by $p(x)$ as well as $p_X(x)$. $\mathbf{x}$ shows the distribution of X in vector form , i.e., $x_i = \mathbf{x}(i) = \mathbb{P}(X = i)$. $n-1$ simplex is the set of points x in n dimensional Euclidean space that satisfy $\sum_i x(i) = 1$. card is the cardinality of a set.

2.3 Causal Model with Minimum Cardinality Exogenous Variable

Consider the causal model $\mathcal{M} = (\{X, Y\}, E_0, f_0, X \to Y, p_{X,E})$. The task is to identify the underlying causal graph $X \to Y$ using independent identically distributed samples $\{(x_i, y_i)\}_i$. Assuming causal sufficiency, this task reduces to deciding whether X causes Y or Y causes X. To isolate the identifiability problem from estimation errors due to finite samples, we assume that the joint distribution of (X, Y) is available. Most proofs are deferred to the Appendix.

One way to identify that X causes Y is by showing that although there exists a function f and random variable E with $Y = f(X, E), X \perp\!\!\!\perp E$, there is no function, random variable pair $(g, \tilde{E})$ such that $X = g(Y, \tilde{E}), Y \perp\!\!\!\perp \tilde{E}$. However, without more assumptions, this is not possible: For any joint distribution one can find valid causal models for both $X \to Y, X \leftarrow Y$. This is widely known, although for completeness, we provide a proof (See Lemma 4 in

the Appendix).

Even when the true causal graph is known, one can create different constructions of f, E with $Y = f(X, E), X \perp\!\!\!\perp E$. There is no way to distinguish the true causal model. However, even though we cannot recover the actual function and the exogenous variable, we can still show identifiability.

First, we give an equivalent characterization of a causal model on two variables.

Definition 1 (**Block Partition Matrices).** *Consider a matrix* $\mathbf{M}$ *such that* $\mathbf{M} \in \{0,1\}^{n^2 \times m}$. *Let* $\mathbf{m}_{i,j}$ *represent the* $i + (j-1)n$ *th row of* $\mathbf{M}$. *Let* $S_{i,j} = \{k \in [m] : \mathbf{m}_{i,j}(k) \neq 0\}$. $\mathbf{M}$ *is called a block partition matrix if it belongs to* $\mathcal{C} \coloneqq \{\mathbf{M} : \mathbf{M} \in \{0,1\}^{n^2 \times m}, \bigcup_{i \in [n]} S_{i,j} = [m], S_{i,j} \cap S_{l,j} = \emptyset, \forall i \neq l\}$.

$\mathcal{C}$ thus stands for $0, 1$ matrices with n^2 rows and m columns where each block of n rows correspond to a partitioning of the set $[m]$. We make the following key observation:

Lemma 1. *Given discrete random variables* X, Y *with distribution* $p(x, y)$, $\exists$ *a causal model* $\mathcal{M} = (\{X, Y\}, E, f, X \to Y, p_{X,E})$, $E \in \mathcal{E}$ *with card*$(\mathcal{E}) = m$ *if and only if* $\exists \mathbf{M} \in \mathcal{C}, \mathbf{e} \in \mathbb{R}_+^m$ *with* $\sum_i \mathbf{e}(i) = 1$ *that satisfy* $vec(\mathbf{Y}|\mathbf{X}) = \mathbf{Me}$.

In other words, the existence of a causal pair $X \to Y$ is equivalent to the existence of a block partition matrix $\mathbf{M}$ and a vector $\mathbf{e}$ of proper dimensions with $vec(\mathbf{Y}|\mathbf{X}) = \mathbf{Me}$.

For simplicity, assume $|\mathcal{X}| = |\mathcal{Y}| = n$. We later remove this constraint. We first show that any joint distribution can be explained using a variable E with $n(n-1)+1$ states.

Lemma 2 (**Upper Bound on Minimum Cardinality of** E)**.** *Let* $X \in \mathcal{X}, Y \in \mathcal{Y}$ *be two random variables with joint probability distribution* $p_{X,Y}(x,y)$*, where* $|\mathcal{X}| = |\mathcal{Y}| = n$*. Then* $\exists$ *a causal model* $Y = f(X,E), X \perp\!\!\!\perp E$ *that induces* $p_{X,Y}$*, where* E *has support size* $n(n-1)+1$*.*

We can show that, if the columns of $\mathbf{Y}|\mathbf{X}$ are uniformly sampled points in the $n-1$ dimensional simplex, then $n(n-1)$ states are also necessary for E (see Proposition 3 in the Appendix). This shows, unless designed by nature through the causal mechanism, exogenous variable cannot have small cardinality. Based on this observation, the hope is to prove that in the wrong causal direction, say $X \to Y$ and we find an $\tilde{E} \perp\!\!\!\perp Y$ such that $X = g(Y, \tilde{E})$ for some g, the exogenous variable $\tilde{E}$ has to have large cardinality. In the next section, we show this is actually through, under mild conditions on f.

2.3.1 Identifiability for H_0 entropy

In a causal system $Y = f(X,E)$, nature chooses the random variables X, E, and function f, and the conditional probability distributions are then determined by these. We are interested in the cardinality of variables $\tilde{E} \perp\!\!\!\perp X$ in the wrong causal direction $X = g(Y, \tilde{E})$. Considering $\mathbf{X}|\mathbf{Y}$, we can show that the same lower bound of $n(n-1)$ still holds despite nature now chooses E and X randomly, rather than choosing the columns of $\mathbf{X}|\mathbf{Y}$ directly. A mild

assumption on f is needed to avoid degenerate cases (For counterexamples see the appendix).

Definition 2 (**Generic Function**). *Let $Y = f(X, E)$ where variables X, Y, E have supports $\mathcal{X}, \mathcal{Y}, \mathcal{E}$, respectively. Let $S_{y,x} = f_x^{-1}(y) \subset \mathcal{E}$ be the inverse map for x, e, i.e., $S_{y,x} = \{e \in \mathcal{E} : y = f(x, e)\}$. A function f is called "generic", if for each (x_1, x_2, y) triple $f_{x_1}^{-1}(y) \neq f_{x_2}^{-1}(y)$ and for every (x, y) pair $f_x^{-1}(y) \neq \emptyset$.*

In other words f is called generic if y^{th} row in the x_1^{th} block of matrix $\mathbf{M}$ in the decomposition $vec(\mathbf{Y}|\mathbf{X}) = \mathbf{Me}$ is different from y^{th} row in the x_2^{th} block, and both are nonzero. This is not a restrictive condition, for example if $p(y|x)$ are all different, no two rows of $\mathbf{M}$ can be the same. For any given conditional distribution, if the probabilities are perturbed by arbitrarily small continuous noise, the corresponding f will be generic almost surely. We have the following main identifiability result:

Theorem 1 (**Identifiability**). *Consider the causal model*

$$\mathcal{M} = (\{X, Y\}, E_0, f_0, X \to Y, p_{X,E_0}),$$

where the random variables X, Y have n states, $E_0 \perp\!\!\!\perp X$ has θ states and f is a generic function .

If the distributions of X and E are uniformly randomly selected from the $n-1$ and $\theta-1$ simplices, then with probability 1, any $\tilde{E} \perp\!\!\!\perp Y$ that satisfies $X = g(Y, \tilde{E})$ for some deterministic function g has cardinality at least $n(n-1)$.

Theorem 1 implies that the causal direction is identifiable, when the exogenous variable has cardinality $< n(n-1)$:

Corollary 1. *Assume that there exists an algorithm $\mathcal{A}$ that given n random variables $\{Z_i\}, i \in [n]$ with distributions $\{p_i\}, i \in [n]$ each with n states, outputs the distribution of the random variable E with minimum cardinality and functions $\{f_i, i \in [n]\}$ where $Z_i = f_i(E)$.*

Consider the causal pair $X \to Y$ where $Y = f(X, E_0)$. Assume that the cardinality of E_0 is less than $n(n-1)$, and f is generic. Then, $\mathcal{A}$ can be used to identify the true causal direction with probability 1, if X, E_0 are selected uniformly randomly from the proper dimensional simplices.

Proof. Feed the set of conditional distributions $\{\mathbb{P}(Y|X=i) : i \in [n]\}$ and $\{\mathbb{P}(X|Y=i) : i \in [n]\}$ to $\mathcal{A}$ to obtain E, $\tilde{E}$. From Theorem 1, with probability 1, $\mathcal{A}$ identifies $\tilde{E}$ with $\text{card}(\tilde{E}) \geq n(n-1)$. Then since $\text{card}(E) \leq \text{card}(E_0) < \text{card}(\tilde{E})$, comparing cardinalities give the true direction. □

Corollary 1 gives an algorithm for finding the true causal direction: Estimate $E, \tilde{E}$ with minimum H_0 entropy and declare $X \to Y$ if $|\tilde{E}| > |E|$ and declare $X \leftarrow Y$ if $|\tilde{E}| < |E|$. The result easily extends to the case where X and Y are allowed to have different number of states:

Proposition 1 (**Inference algorithm**). *Suppose $X \to Y$. Let $X \in \mathcal{X}, Y \in \mathcal{Y}$, $|\mathcal{X}| = n, |\mathcal{Y}| = m$. Assume that $\mathcal{A}$ is the algorithm that finds the exogenous variables $E, \tilde{E}$ with minimum cardinality. Then, if the underlying exogenous*

variable E_0 *satisfies* $|E_0| < n(m-1)$*, with probability 1, we have* $|X| + |E| < |Y| + |\tilde{E}|$.

Proof follows from Corollary 1, and by extending the proof of Theorem 1 to different cardinalities for X, Y.

Unfortunately, it turns out there does not exist an efficient algorithm $\mathcal{A}$, unless P=NP:

Theorem 2. *Given a conditional distribution matrix* $\mathbf{Y}|\mathbf{X}$*, identifying* $E \perp\!\!\!\perp X$ *with minimum support size such that there exist a function* f *with* $Y = f(X, E)$ *is* NP *hard.*

The hardness of this problem sets us to search for alternative approaches.

2.4 Causal Model with Minimum H_1 Entropy

In this section, we propose a way to identify the causal model that explains the observational data with minimum Shannon entropy (entropy in short). Entropy of a causal model is measured by the number of random bits required to generate its input. In the causal graph $X \to Y$, where $Y = f(X, E)$, we identify the exogenous variable $E \perp\!\!\!\perp X$ with minimum entropy. We show that this corresponds to a known problem which has been shown to be NP hard. Later we propose a greedy algorithm.

Notice that $H(E)$ is different from the conditional entropy $H(Y|X)$. Certainly, since $Y = f(X, E)$, $H(Y|X) \leq H(E)$. The key is that since E is

forced to be independent from X, $H(E)$ cannot be lowered to $H(Y|X)$. To see this, we can write $H(Y|X) = \sum_i p_X(i)H(Y|X=i)$, whereas since conditional probability distribution of $Y|X=i$ is the same as the distribution of $f_i(E)$ for some function f_i, we have $H(E) \geq \max_i H(Y|X=i)$.

2.4.1 Finding E with minimum entropy

Consider the equation $Y = f(X,E), X \perp\!\!\!\perp E$. Let $f_x : \mathcal{E} \to \mathcal{Y}$ be the function mapping E to Y when $X = x$, i.e., $f_x(E) \coloneqq f(x,E)$. Then $\mathbb{P}(Y=y|X=x) = \mathbb{P}(f_x(E)=y|X=x) = \mathbb{P}(f_x(E)=y)$. The last equality follows from the fact that $X \perp\!\!\!\perp E$. Thus, we can treat the conditional distributions $\mathbb{P}(Y|X=x)$ as distributions that emerge by applying some function f_x to some unobserved variable E. Then the problem of identifying E with minimum entropy given the joint distribution $p(x,y)$ becomes equivalent to, given distributions of the variables $f_i(E)$, finding the distribution with minimum entropy (distribution of E), such that there exists functions f_i which map this distribution to the observed distributions of $Y|X=i$. It can be shown that $H(E) \geq H(f_1(E), f_2(E), \ldots, f_n(E))$. Regarding $f_i(E)$ as a random variable U_i, the best lower bound on $H(E)$ can be obtained by minimizing $H(U_1, U_2, \ldots, U_n)$. We can show that we can always construct an E that acheives this minimum. Thus the problem of finding the exogenous variable E with minimum entropy given the joint distribution $p(x,y)$ is equivalent to the problem of finding the minimum entropy joint distribution of the random variables $U_i = (Y|X=i)$, given the marginal distributions $p(Y|X=i)$:

Theorem 3 (**Minimum Entropy Causal Model**). *Assume that there exists an algorithm $\mathcal{A}$ that given n random variables $\{Z_i\}, i \in [n]$ with distributions $\{p_i\}, i \in [n]$ each with n states, outputs the joint distribution over Z_i consistent with the given marginals, with minimum entropy.*

Then, $\mathcal{A}$ can be used to find the causal model $\mathcal{M} = (\{X, Y\}, E, X \to Y, p_{X,E})$ with minimum input entropy, given any joint distribution $p_{X,Y}$.

The problem of minimizing entropy subject to marginal constraints is non-convex. In fact, it is shown in [75] that minimizing the joint entropy of a set of variables given their marginals is NP hard. Thus we have the following corollary:

Corollary 2. *Finding the causal model $\mathcal{M} = (\{X, Y\}, E, f, X \to Y, p_{E,X})$ with minimum $H(E)$ that induce a given distribution $p(x, y)$ is* NP *hard.*

For this, we propose a greedy algorithm. Using entropy to identify E instead of cardinality, despite both turning out to be NP hard, is useful since entropy is more robust to noise in data. In real data, we estimate the probability values from samples, and noise is unavoidable.

2.4.2 A Conjecture on Identifiability with H_1 Entropy

We have the following conjecture, supported by artificial and real data experiments in Section 2.5.

Conjecture 1. *Consider the causal model $\mathcal{M} = (\{X, Y\}, E, f, X \to Y, p_{X,E})$ where discrete random variables X, Y have n states, $E \perp\!\!\!\perp X$ has θ states.*

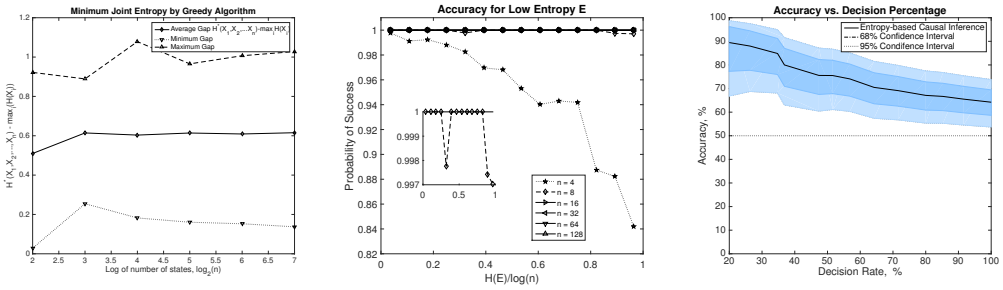
If the distribution of X is uniformly randomly selected from the $n-1$ dimensional simplex and distribution of E is uniformly selected from the probability distributions that satisfy $H_1(E) \leq \log n + \mathcal{O}(1)$ and f is randomly selected from all functions $f : [n] \times [\theta] \to [n]$, then with high probability, any $\tilde{E} \perp\!\!\!\perp Y$ that satisfies $X = g(Y, \tilde{E})$ for some deterministic g entails $H(X) + H(E) < H(Y) + H(\tilde{E})$.

Proposition 2 (Assuming Conjecture 1)**.** *Assume there exists an algorithm $\mathcal{A}$ that given n random variables $\{Z_i\}, i \in [n]$ with distributions $\{p_i\}, i \in [n]$ each with n states, outputs the distribution of the random variable E with minimum entropy and functions $\{f_i\}, i \in [n]$ where $Z_i = f_i(E)$.*

Consider the causal pair $X \to Y$ where $Y = f(X, E_0)$, and cardinality of E_0 is cn for some constant c, and f is selected randomly. Then, $\mathcal{A}$ can be used to identify the true causal direction with high probability, if X, E_0 are uniformly random samples from the proper dimensional simplices.

2.4.3 Greedy Entropy Minimization Algorithm

Given m discrete random variables with n states, we provide a heuristic algorithm to minimize their joint entropy given their marginal distributions. The main idea is the following: Each marginal probability constraint must be satisfied. For example, for the case of two variables with distributions p_1, p_2, ith row of joint distribution matrix should sum to $p_1(i)$. The contribution of a probability mass to the joint entropy only increases when probability mass is divided into smaller chunks: $-p_1(i) \log p_1(i) \leq -a \log a - b \log b$, when



(a) Greedy Entropy Min. (b) Identifiability (c) Real Data

Figure 2.1: (a) Performance of greedy joint entropy minimization algorithm: n distributions each with n states are randomly generated for each value of n. As can be seen, the minimum joint entropy obtained by the greedy algorithm is at most 1 bit away from the largest marginal $\max_i H(X_i)$. (b) Identifiability with Entropy: We generate distributions of X, Y by randomly selecting f, X, E. Probability of success is the fraction of points where $H(X, E) < H(Y, \tilde{E})$. As observed, larger n drives probability of success to 1 when $H(E) \leq \log n$, supporting Conjecture 1. (c) Real Data Performance: Decision rate is the fraction of samples for which algorithm makes a decision for a causal direction. A decision is made when $|H(X, E) - H(Y, \tilde{E})| > t \log_2 n$, where t determines the decision rate. Confidence intervals are also provided.

$p_1(i) = a + b$, for $a, b \geq 0$. Thus, we try to keep large probability masses intact to assure that their contribution to the joint distribution is minimized.

We propose Algorithm 1. The sorting step is only to simplify the presentation. Hence, although the given algorithm runs in time $\mathcal{O}(m^2 n^2 \log n)$, it can easily be reduced to $\mathcal{O}(\max(mn \log n, m^2 n))$ by dropping the sorting step. The algorithm simply proceeds by removing the most probability mass it can at each round. This makes sure the large probability masses remain intact.

One can easily construct the joint distribution using a variant: Instead of sorting, at each step, find $r = \min_i\{\max_j\{p_i(j)\}\}$ and assign r to the element

Algorithm 1 Joint Entropy Minimization Algorithm

Input: Marginal distributions of m variables each with n states, in matrix form $\mathbf{M} = [p_1^T; p_2^T; ..., p_m^T]$.
$e = [\ ]$
Sort each row of M in decreasing order.
Find minimum of maximum of each row: $r \leftarrow \min_i(p_i(1))$
while $r > 0$ **do**
    $e \leftarrow [e, r]$
    Update maximum of each row: $p_i(1) \leftarrow p_i(1) - r, \forall i$
    Sort each row of M in decreasing order.
    $r \leftarrow \min_i(p_i(1))$
end while
Return e.

with coordinates (a_i), where $a_i = \arg\max_j p_i(j)$.

Lemma 3. *Greedy entropy minimization outputs a point with entropy at most* $\log m + \log n$.

Lemma 3 follows from the fact that the algorithm returns a support of size at most $m(n-1)+1$.

We also prove that, the algorithm returns a point that satisfies the KKT conditions of the optimization problem, which implies that it is a local optimum (see Theorem 1 in [70]).

2.5 Experiments

In this section, we test the performance of our algorithms on real and artificial data. First, we test the greedy entropy minimization algorithm and show that it performs close to the trivial lower bound. Then, we test

our conjecture of identifiability using entropy. Lastly, we test our entropy-minimization based causal identification technique on real data.

In order to test our algorithms, we sample points in proper dimensional simplices, which correspond to distributions for X and E. Distribution of points are uniform for selecting the distribution of X. It is well-known that a vector $[x_i/Z]_i$ is uniformly randomly distributed over the simplex, if x_i are i.i.d. exponential random variables with parameter 1, and $Z = \sum_i x_i$ [102]. To sample low-entropy distributions for E, instead of exponential, we use a heavy tailed distribution for sampling each coordinate. Specifically, we use $[e_i/Z]_i$, where e_i are i.i.d. log-normal random variables with parameter σ. We observe that this allows us to sample a variety of distributions with small entropy.

Performance of Greedy Entropy Minimization: We sample distributions for n random variables $\{X_i\}, i \in [n]$ each with n states and apply Algorithm 1 to minimize their joint entropy. We compare our greedy joint entropy minimization algorithm with the simple lower bound of $\max_i H(X_i)$. Figure 2.1a shows average, maximum and minimum excess bits relative to this lower bound. Contrary to the pessimistic bound of $\log n$ bits, joint entropy is at most 1 bit away from $\max_i H(X_i)$ for the given range of n.

Verifying Entropy-Based Identifiability Conjecture: In this section, we empirically verify Conjecture 1. The distributions for X are uniformly randomly sampled from the simplex in n dimensions. We also select f randomly (see implementation details). For the log-normal parameter σ used for sampling the distribution of E from the $n(n-1)$ dimensional simplex, we sweep the

integer values from 2 to 8. This allows us to get distribution samples from different regimes. We only consider the samples which satisfy $H(E) \leq \log n$.

After sampling E, X, f, we identify the corresponding $\mathbf{Y}|\mathbf{X}$ and $\mathbf{X}|\mathbf{Y}$ for $Y = f(X, E)$. We apply greedy entropy minimization on the columns of the induced distributions $\mathbf{Y}|\mathbf{X}, \mathbf{X}|\mathbf{Y}$ to get the estimates $E, \tilde{E}$ for both causal models $Y = f(X, E)$ and $X = g(Y, \tilde{E})$, respectively. Figure 2.1b shows the variation of success probability, i.e., the fraction of samples which satisfy $H(X) + H(E) < H(Y) + H(\tilde{E})$. As observed, as n is increased, probability of success converges to 1, when $H(E) \leq \log n$, which supports the conjecture.

Experiments on Real Cause Effect Pairs: We test our entropy-based causal inference algorithm on the CauseEffectPairs repository [96]. ANM have been reported to achieve an accuracy of 63% with a confidence interval of $\pm 10\%$ [98]. We also use the binomial confidence intervals as in [25].

The cause effect pairs show very different characteristics. From the scatter plots, one can observe that they can be a mix of continuous and discrete variables. The challenge in applying our framework on this dataset is choosing the correct quantization. Small number of quantization levels may result in loss of information regarding the joint distribution, and a very large number of states might be computationally hard to work with. We pick the same number of states for both X and Y, and use a uniform quantization that assures each state of the variables has ≥ 10 samples on average. From the samples, we estimate the conditonal transition matrices $\mathbf{Y}|\mathbf{X}$ and $\mathbf{X}|\mathbf{Y}$ and feed the columns to the greedy entropy minimization algorithm (Algorithm 1), which outputs

an approximate of the smallest entropy exogenous variable. Later we compare $H(X, E)$ and $H(Y, \tilde{E})$ and declare the model with smallest input entropy to be the true model, based on Conjecture 1.

For a causal pair, we invoke the algorithm if $|H(X, E) - H(Y, \tilde{E})| \geq t \log(n)$ for threshold parameter t, which determines the decision rate. Accuracy becomes unstable for very small decision rates, since the number of evaluated pairs becomes too small. At 100% decision rate, algorithm achieves 64.21% which is slightly better than the 63% performance of ANM as reported in [98]. In addition, our algorithm only uses probability values, and is applicable to categorical as well as ordinal variables.

Chapter 3

Entropic Latent Variable Discovery

We consider the problem of discovering the simplest latent variable that can make two observed discrete variables conditionally independent. This problem has appeared in the literature as probabilistic latent semantic analysis (pLSA), and has connections to non-negative matrix factorization. When the simplicity of the variable is measured through its cardinality (Renyi entropy with $\alpha = 0$), we show that a solution to this latent variable discovery problem can be used to distinguish direct causal relations from spurious correlations among almost all joint distributions on simple causal graphs with two observed variables. Conjecturing a similar identifiability result holds with Shannon entropy, we study a loss function that trades-off between entropy of the latent variable and the conditional mutual information of the observed variables. We then propose a latent variable discovery algorithm – *LatentSearch* – and show that its stationary points are the stationary points of our loss function. We experimentally show that LatentSearch can indeed be used to distinguish direct causal relations from spurious correlations.

Consider the following problem: we are given two discrete random

The author of this dissertation contributed to the conception of the research problem, theoretical developments and experimental validation.

variables X, Y over m and n states respectively. Suppose we want to construct a third random variable Z, such that X and Y are independent conditioned on Z. Without any constraints this can be trivially achieved: Simply picking $Z = X$ or $Z = Y$ ensures that $X \perp\!\!\!\perp Y \,|Z$. However, this requires that the random variable Z is as *complex* as X or Y. We therefore ask the following question: *is there a simple Z that makes X, Y conditionally independent?* Suppose we measure the complexity of Z by its cardinality (equivalently, its Renyi entropy for $\alpha = 0$). Then, a non-trivial answer to this question would require us to find a random variable Z with k states, where $k < \min\{m, n\}$ that renders X, Y conditionally independent. This problem of recovering a small cardinality latent variable Z is closely related to Probabilistic Latent Semantic Analysis (pLSA), non-negative matrix factorization (NMF) and Latent Dirichlet Allocation (LDA). In this paper we show that a solution to this problem can also be used for causal inference in the presence of latent variables: Suppose, given two dependent random variables X, Y, that our objective is to decide if there is a direct causal relation between the two, or if the observed correlation is spurious, i.e., due to an indirect path via a latent variable.

Our main theoretical result is that it is always possible to distinguish the latent causal graph from the triangle causal graph (see Figure 3.1) where Z is latent, if the latent variable has *cardinality (aka Renyi entropy for $\alpha = 0$)* less than $\min\{m, n\}$, except for a measure zero set of joint distributions. We also study this problem using an alternate simplicity metric of *Shannon entropy* instead of cardinality. In this case, our problem becomes: Given two discrete

variables X, Y with m, n states, respectively, what is the minimum entropy variable Z with k states that assures $X \perp\!\!\!\perp Y \,|Z$? We conjecture that an identifiability result also holds in this case even when $k \geq n$, thus permitting us to study a larger class of latent Z. More precisely, we expect most distributions generated from the triangle graph with small entropy Z to require a large entropy Z to satisfy conditional independence $X \perp\!\!\!\perp Y \,|Z$.

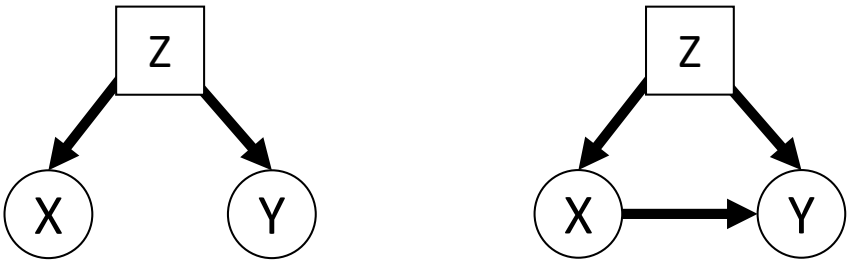
With these observations, our main algorithmic contribution is as follows: To solve the problem of recovering the minimum entropy Z that renders the observed variables X, Y conditionally independent, we propose an algorithm to minimize the loss $I(X;Y|Z) + \beta H(Z)$. The parameter β allows us to discover a tradeoff between *the simplicity of variable* Z and *how much of the dependence between* X, Y *it can explain away.* We show that our algorithm always outputs a stationary point of the loss function. Moreover, for $\beta = 1$, we are able to show convergence of the algorithm, and that it converges to either a local minimum or a saddle point. We also empirically demonstrate that it converges much faster than gradient descent and recovers better solutions after convergence.

Our contributions are as follows:

- Given a joint dist. between two discrete variables X, Y, we propose *LatentSearch* – a latent variable discovery algorithm that constructs a third variable Z to minimize $I(X;Y|Z) + \beta H(Z)$.
- We show that the stationary points of our algorithm are also stationary points of the loss $I(X;Y|Z) + \beta H(Z)$. For $\beta = 1$, we prove convergence and show that the algorithm converges to either a local minimum or a

saddle point.

- Using our algorithm, we empirically demonstrate a fundamental tradeoff between the discovered latent variable's complexity, measured by its entropy, and how much of the dependence between the observed variables it can explain away.
- We show that if we are given an algorithm that recovers the variable Z with minimum number of states, this algorithm can be used to distinguish the two causal graphs $X \leftarrow Z \rightarrow Y$ from $X \leftarrow Z \rightarrow Y, X \rightarrow Y$, where Z is latent. We conjecture that the identifiability result holds if we use Shannon entropy of the variable, instead of its cardinality.
- We use our latent variable discovery algorithm to test the validity of our conjecture on simulated data and show that the true causal graph is identifiable even when $k \geq n$ with probability close to 1.
- Our latent variable discovery algorithm can be used to answer the question: *Is the causal relation between two given variables direct, or is there a simple variable conditioned on which, they become independent.* Based on this, we use our algorithm on Adult dataset from UCI repository [28] to recover the skeleton of the causal graph. We show that, for a carefully chosen entropy threshold for the unobserved variables, our algorithm recovers almost the exact skeleton as the BIC score based structure learning algorithm.



(a) Latent graph (b) Triangle graph

Figure 3.1: Causal graphs we want to distinguish. Z is latent (unobserved).

3.1 Background and Notation

Let $D = (\mathcal{V}, E)$ be a directed acyclic graph on the set of vertices $\mathcal{V} = \{V_1, V_2, \ldots, V_n\}$ with directed edge set E. Each directed edge is a tuple (V_i, V_j), where $V_i, V_j \in \mathcal{V}$. Let $\mathcal{P}$ be a joint distribution over a set of variables labeled by $\mathcal{V}$. D is called a valid Bayesian network for the distribution $\mathcal{P}$ is the distribution factorizes with respect to the graph as $\mathcal{P}(V_1, V_2, \ldots V_n) = \prod_i \mathcal{P}(V_i|pa_i)$, where pa_i are the set of parents of vertex V_i in graph D. If D is a valid Bayesian network for $\mathcal{P}$, if the three vertices X, Y, Z satisfy a purely graphical criterion called the *d-separation*, $X \perp\!\!\!\perp Y \,|Z$ with respect to $\mathcal{P}$. A distribution $\mathcal{P}$ is called faithful to graph D if the converse is also true: Any three variables such that $X \perp\!\!\!\perp Y \,|Z$ satisfy the d-separation criterion on graph D.

Note that the edges in a Bayesian network do not carry a physical meaning: They simply indicate how a joint distribution can be factorized. *Causal Bayesian networks* (or causal graphs[1]) extend the notion of Bayesian

[1]In this work, we do not use structural causal models, hence causal graph refers to causal Bayesian networks.

networks to different experimental, the so called interventional settings. *An intervention* is an experiment that changes the workings of the underlying system and sets the value of a variable, shown as $do(X = x)$. Causal Bayesian networks allow us to calculate the joint distributions under these experimantal conditions, called the interventional distributions [2].

In this paper, we will work with the simple causal graphs given in Figure 3.1. From the d-separation principle, we see that the *latent* graph satisfies $X \perp\!\!\!\perp Y \,|Z$, whereas under the faithfulness condition, $X \not\perp\!\!\!\perp Y \,|Z$ in the *triangle graph*. Checking the existence of such a latent variable can help us recover the true causal graph as we discover in the next sections.

In this paper, we work with discrete ordinal or categorical variables. Suppose the cardinalities of the observed variables X, Y are m, n, respectively. The joint distribution can be then be represented with an $m \times n$ non-negative matrix whose entries sum to 1. We assume that we have knowledge of this joint distribution (of the pair of observed variables).

We use $[n]$ to represent the set $\{1, 2, \ldots, n\}$ for any positive integer n. Capital letters are used to represent random variables whereas lowercase letters are used to represent realizations unless otherwise stated [3]. Letters X, Y are reserved to represent the observed variables, whereas letter Z is used to represent the latent variable. To represent the probability mass function

[2] Due to space constraints, we cannot formally introduce Pearl's framework. Please see [126, 105].

[3] In the supplementary material, x_i is used to represent the probability that variable X takes the value i.

over three variables X, Y, Z, we use $p(x, y, z) \coloneqq \mathbb{P}(X = x, Y = y, Z = z)$ and similarly for any conditional $p(z|x, y) \coloneqq \mathbb{P}(Z = z|X = x, Y = y)$. Lowercase boldface letters are used for vectors and uppercase boldface letters are used for matrices. We also use $p(Z|x, y)$ to represent the conditional probability mass function $\mathbb{P}(Z|X = x, Y = y)$ (Similarly for $p(Z|x), p(Z|y)$). $\text{card}(X)$ stands for the cardinality of the size of X's support. *Entropy* of a random variable X, shown via $H(X)$, refers to its Shannon entropy, measured as $H(X) = -\sum_x p(x) \log(p(x))$.

3.2 Related Work

Tools for Latent Variable Discovery: Latent variables have been used to model and explain dependence between observed variables in different communities under different names. One of the first models is the *probabilistic latent semantic analysis (pLSA)* framework developed by [53]. Given two variables X, Y pLSA constructs a variable Z that makes the two variables conditionally independent via two equivalent formulations. These two formulations correspond to fitting either the Bayesian network $X \to Z \to Y$ or $X \leftarrow Z \to Y$. pLSA is solved by running EM algorithm. However it does not have an incentive to discover low-entropy latents. In Section 3.5, we will see that running EM on top of our algorithm cannot help discover better solutions for our problem.

Latent Dirichlet allocation (LDA) is another framework for latent variable discovery, which has been widely used especially in topic modeling [15, 8]. It is developed for creating a generative model of a given set of documents.

Given m documents each with a collection of words out of a dictionary with n words, LDA recovers a factorization of the word count matrix $\mathbf{M} = \mathbf{UV}$, where the columns of V are the mixtures of topics for each document. When the prior of these mixtures is chosen from a Dirichlet distribution with parameter less than 1, it encourages low entropy columns for V. However notice that when we map LDA to our setting, this does not correspond to the entropy of the latent variable: The distribution of the latent variable can be obtained from the LDA decomposition as the row sums of the matrix V, which can have large entropy even if columns of V are 1 sparse.

Non-negative matrix factorization is another approach that can be seen as explaining the observed dependence between two variables with a simple latent factor. Compared to PCA, NMF explains the data with non-negative factors. Given a matrix $\mathbf{M}$, NMF approximates it as $\mathbf{M} \approx \mathbf{UDV}$, where $\mathbf{U}, \mathbf{D}, \mathbf{V}$ are m-by-k,k-by-k,k-by-n non-negative matrices, respectively and $\mathbf{D}$ is a diagonal matrix. Factorizing the joint distribution matrix between two observed variables via NMF with generalized KL divergence loss recovers solutions to the pLSA problem [41]. NMF, in general, is not unique.

Perhaps the most relevant to ours are the two papers in the Bayesian setting [18, 123]. They use low-entropy priors on the latent variable's distribution while performing inference. However their approach is different and their methods cannot be used to discover the tradeoff between unexplained away dependence and complexity of the latent variable as we do. In [123], the authors use low-entropy prior as a proxy for discovering latent factors with

sparse support.

Our approach is also related to the *information bottleneck principle*. There, the authors propose an iterative algorithm to construct the conditional probability distribution of $Z|X$ for the Markov chain $Y \to X \to Z$, where X, Y are observed. Their objective is to make sure Z is as informative as possible about X, while being the least informative about Y. Accordingly, they propose an algorithm that after convergence outputs a stationary point of the loss function $\mathcal{L} = I(X;Z) - \lambda I(Y;Z)$.

Learning Causal Graphs with Latents: Learning causal graphs with and without latent variables has been extensively studied in the literature. In graphs with many observed variables, although many of the causal edges can be recovered from the observational data (for example through algorithms that employ CI (conditional independence) tests such as IC [105], PC [126]), some of the causal edges are not identifiable unless one performs experiments and collects additional data. The existence of latent variables make the problem even harder. Although extensions of these algorithms can be employed (IC*, FCI and derivatives), in general, fewer number of edges can be learned due to the confounding that cannot be controlled for: Latent variables make the CI tests less informative, by inducing spurious correlations between the observed variables. Especially, even in the simple case when we want to decide between the latent and triangle graphs in Figure 3.1, these CI based algorithms cannot be used.

Our approach is, in essence, similar to [69]. There, the authors assume

that the unobserved background variables have small entropy and suggest an algorithm to distinguish between the causal graphs $X \to Y$ and $X \leftarrow Y$. Our setup is also similar to the one in [63], where they identify a condition on the conditional distribution $p(Y|X)$ that, if true, implies that there does not exist any simple latent variable Z that can make X, Y conditionally independent. This assumption, in the discrete variable setup, roughly implies that the matrix $p(X, Y)$ is sparse in a structured way. In the continuous variable setting, [119] propose using kernel methods combined with a rank argument to detect latent confounders between two variable. [139] analyzes the discoverability of causal structures with latents using the entropic vector of the variables. Finally, related work also includes [62] and [77], where the authors extend the additive noise model based approach in [54] to the case with a latent confounder.

3.3 LatentSearch: An Algorithm for Latent Variable Discovery

We formulate the problem of discovering the latent variable with small entropy that explains much of the observed dependence. The remaining dependence between the two observed variables X, Y (which is unexplained by Z) is measured using conditional mutual information $I(X; Y|Z)$. The suitability of this metric stems from the fact that $I(X; Y|Z) = 0 \iff X \perp\!\!\!\perp Y \,|Z$. The smaller Z makes $I(X; Y|Z)$, the more dependence it explains. While it is easy to construct a Z that makes $I(X; Y|Z) = 0$, our goal is to discover a *simple* latent variable Z. To quantify the simplicity of the latent variable, we use its

Shannon entropy $H(Z)$. Moreover, we would like to allow a tradeoff between these two factors, namely the simplicity of the latent variable and the residual dependency between X, Y after conditioning on Z. Accordingly, we introduce the loss function

$$\mathcal{L} = I(X;Y|Z) + \beta H(Z). \tag{3.1}$$

Recall in our setting that we have access of $p(x, y)$, the distribution between the discrete *observed* random variables (X, Y) (with $\text{card}(X) = m$ and $\text{card}(Y) = n$). Thus, constructing a latent variable Z is equivalent to constructing a joint distribution $q(x, y, z)$ between three variables (X, Y, Z), that respects the observed joint between (X, Y), i.e., $p(x, y)$. This leads to the requirement that $\sum_z q(x, y, z) = p(x, y)$. Rather than optimizing for $q(x, y, z)$ and forcing this constraint, we simply optimize for $q(z|x, y)$ and set $q(x, y, z) = q(z|x, y)p(x, y)$. Therefore we have $\mathcal{L} = \mathcal{L}(q(z|x, y))$.

Additionally, the cardinality of Z is one measure of simplicity of Z, and further determines the number of variables we optimize to minimize the loss $\mathcal{L}$. If $\text{card}(Z) = k, \text{card}(X) = m, \text{card}(Y) = n$, to describe the conditional $q(z|x, y)$ we need kmn non-negative numbers. Moreover, we have the constraint that $\sum_z q(z|x, y) = 1, \forall x, y$.

To this end, we propose *Algorithm LatentSearch* that starting with the known $p(x, y)$, iteratively computes $q(z|x, y)$ that optimizes to minimize the loss (3.1). Specifically, it marginalizes the joint $q_i(x, y, z)$ to get $q_i(z|x), q_i(z|x)$, and $q_i(z)$, and imposing a scaled product form on these marginals, updates the joint to return $q_{i+1}(x, y, z)$. This is formally shown in Algorithm 2. This

decomposition and update is motivated by the partial derivatives associated with the Lagrangian of the loss function (3.1), and has the formal properties described below. The proofs are delegated to the supplementary material.

Algorithm 2 LatentSearch: Iterative Update Algorithm

Input: Supports of $x, y, z, \mathcal{X}, \mathcal{Y}, \mathcal{Z}$, respectively and a parameter $\beta > 0$. Observed joint $p(x, y)$. Initialization $q_1(z|x, y)$. Number of iterations N.
Output: Joint distribution $q(x, y, z)$
for $i \in [N]$ **do**
Form the joint and marginalize:
$q_i(x, y, z) = q_i(z|x, y)p(x, y) \forall x, y, z.$
$q_i(z|x) = \frac{\sum_{y \in \mathcal{Y}} q_i(x,y,z)}{\sum_{y \in \mathcal{Y}, z \in \mathcal{Z}} q_i(x,y,z)}.$
$q_i(z|y) = \frac{\sum_{x \in \mathcal{X}} q_i(x,y,z)}{\sum_{x \in \mathcal{X}, z \in \mathcal{Z}} q_i(x,y,z)}.$
$q_i(z) = \sum_{x \in \mathcal{X}, y \in \mathcal{Y}} q_i(x, y, z).$
Update:
$q_{i+1}(z|x, y) = \frac{1}{N(x,y)} \frac{q_i(z|x)q_i(z|y)}{q_i(z)^{1-\beta}}$, where $N(x, y) = \sum_{z \in \mathcal{Z}} \frac{q_i(z|x)q_i(z|y)}{q_i(z)^{1-\beta}}$.
end for
return $q_{N+1}(z|x, y)p(x, y)$

Theorem 4. *The stationary points of LatentSearch are also the stationary points of the loss function in (3.1).*

Theorem 5. *For $\beta = 1$, LatentSearch converges to either a local minimum or a saddle point of the loss function in (3.1).*

LatentSearch can be run a number of times by varying the value of β to discover what we believe is a fundamental $I(X; Y|Z)$ vs $H(Z)$ tradeoff curve. An example is given in Figure B.1 in the supplementary material. We believe

the algorithm finds an approximation to a fundamental tradeoff curve such that no point below this curve can be achieved for a given joint distribution $p(x, y)$.

3.4 Detecting Spurious Correlations

Consider the two causal graphs given in Figure 3.1. Suppose there exists an algorithm that can answer the question *"Is there a simple latent variable that can make X, Y conditionally independent?"*. Then we can use the such an algorithm to recover the true graph: If the answer is *yes* (i.e. there is a simple Z that renders (X, Y) to be conditionally independent), then declare the *latent graph* to be the true graph. If the answer is no, declare the *triangle graph* to be the true graph.

Now suppose that the joint distribution comes truly from the triangle graph, and we have access only to $p(x, y)$. Then, if $k > n$, it is possible to construct a latent graph that is indistinguishable from the triangle graph, i.e. agrees over the observed over $p(x, y)$. This is trivially possible for instance by setting $Z = X$ or $Z = Y$. Below, we show that when $k < \min\{m, n\}$, then for any random instance (construction detailed in Theorem 6) of the triangle graph[4], we cannot construct such a low-cardinality Z (equivalently, the simple test fails only on a measure zero set of joint distributions).

[4]If (X, Y, Z) were fully observable, then it is known that among all distributions that are Markov with respect to a given graph, all but a measure zero set are faithful [91]. Therefore, except for a measure zero set, $X \not\perp\!\!\!\perp Y \,|Z$ for distributions from the triangle graph. However note that we do not observe Z and ask the question *does there exist such a Z?*

Theorem 6. *Consider three discrete random variables X, Y, Z with supports $[m], [n], [k]$, respectively, where $k < \min\{m, n\}$. Let $p(x, y, z)$ be the joint distribution over X, Y, Z. Consider the following generative model for $p(x, y, z)$:*

Let the conditional distributions $p(Z), p(X|z), p(Y|x, z)$ be sampled independently and uniformly randomly from the probability simplex in the appropriate dimensions, for any realizations $(z, x) \in [k] \times [m]$.

Then, with probability 1, there does not exist any joint distribution $q(x, y, z)$ such that $\sum_z q(x, y, z) = p(x, y)$ and $X \perp\!\!\!\perp Y \,|Z$.

Corollary 3. *For almost all joint distributions that are obtained for* triangle graph *in Figure 3.1, there does not exist a joint distribution $q(x, y, z)$ that can be encoded by the* latent graph*.*

Proof. The statement follows from the fact that the proposed generative model induces a non-zero probability measure on every joint distribution, which is the set of distributions that can be encoded from the *triangle graph* and any distribution that can be encoded by the *latent graph* requires $X \perp\!\!\!\perp Y \,|Z$, which we have shown happens with probability zero. □

We expect a similar claim to hold, had we used Shannon entropy instead of cardinality. If true, this would allow us to consider latents with cardinality greater than n, but with their *Shannon entropy being small* (formalized below). This can be implemented using *LatentSearch* (Algorithm 2) as a black box for recovering the simplest latent variable. The corresponding algorithm is given in Algorithm 3 (denoted henceforth as *InferGraph*).

Suppose the latent factor has n states, i.e., the same number of states as Y. When the probability values are in general position in the sense described in Theorem 6 and the model comes from the *triangle graph*, we expect that no matter what the decomposition is, the latent variable's entropy cannot be much less than the entropies of X, Y. Motivated by this, we propose a conjecture that formalizes the claim that most of the joint distributions from the *complete graph* factorize only with latents with entropy close to $\min\{H(X), H(Y)\}$.

Conjecture 2. *Consider the following generative model for the joint distribution $p(x, y)$ over two discrete variables X, Y with m, n states, respectively:*

Let the distribution of Z be sampled from Dirichlet(α) for some $\alpha \leq 1$. Let each conditional distribution for the triangle graph, i.e., $\mathbb{P}(X|z), \mathbb{P}(Y|x, z)$ be sampled with Dirichlet(1), i.e., uniformly from the simplex, for all x, z.

Then there are constants a, b such that with probability 1, any joint distribution $q(x, y, z)$ where $\sum_z q(x, y, z) = p(x, y)$ and $X \perp\!\!\!\perp Y \,|Z$ satisfies $H(Z) \geq \theta = a \min(H(X), H(Y)) - b$.

3.5 Simulations

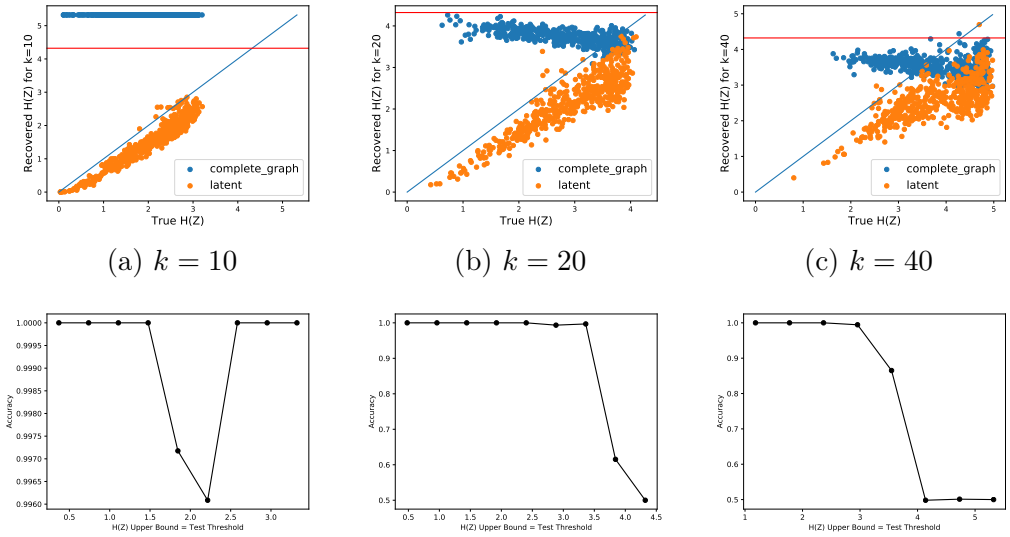
3.5.1 Synthetic Data

In this section, our objective is to distinguish the two causal graphs given in Figure 3.1. Figure 3.2 shows how the recovered latent variable's entropy varies with the entropy of the true latent variable, when the causal graph is the latent graph shown in orange and the triangle graph (see Figure 3.1) shown

Algorithm 3 InferGraph: Causal Inference Algorithm

Input: k : Cardinality of the latent variable Z to be constructed. Observed joint $p(x, y)$ over the variables X, Y. θ : Threshold for $H(Z)$. T : Conditional mutual information threshold.
Initialize a set of N conditional distributions $q_0^i(z|x, y), i \in [N]$.
for $i \in [N]$ **do**
  $q^i(z|x, y) \leftarrow$ *LatentSearch* (Algorithm 2) with input $q_0^i(z|x, y)$.
  Calculate $I^i(X; Y|Z)$ and $H^i(Z)$ from $q^i(x, y, z) = q^i(z|x, y)p(x, y)$.
end for
$S = \{i : I^i(X; Y|Z) \leq T\}$.
$h = \min(\{H^i(Z) : i \in S\})$
if $h > \theta$ **then**
  $D = X \leftarrow Z \rightarrow Y, X \rightarrow Y$
else if $h \leq \theta$ **then**
  $D = X \leftarrow Z \rightarrow Y$
end if
return Causal Graph D.

in blue. The scatter plots are obtained for conditional mutual information threshold of 0.001: For each sampled causal model (each dot in the scatter plots), we randomly initialized 40 conditional distributions for different values of β in (3.1) and run *LatentSearch* (Algorithm 2) for 1000 iterations. Each conditional distribution is chosen uniformly randomly from the corresponding simplex. After *LatentSearch* terminates, of all the 40 recovered distributions we picked the one with minimum entropy of all those that satisfy $I(X; Y|Z) \leq 0.001$. To obtain the scatter plots, for each causal graph, we repeated the above procedure 500 times for each graph, where each 125 of 500 samples are obtained from the Dirichlet distribution with parameters $[1.0, 0.5, 0.2, 0.1]$. This lets us induce a more uniform distribution on the entropy $H(Z)$ (hence the contiguity of samples in x axis). Finally, for each point in the scatter plot, we predicted the

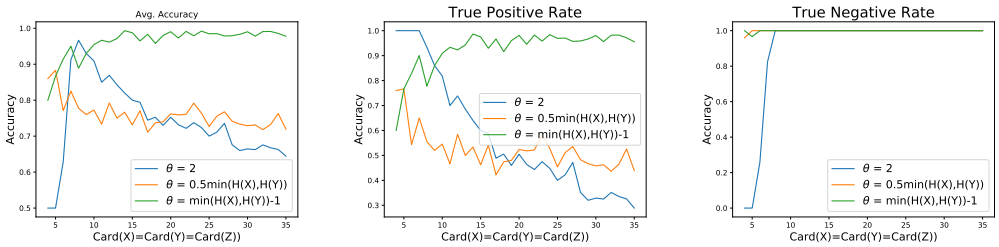


(a) $k = 10$ (b) $k = 20$ (c) $k = 40$

(d) $k = 10$ (e) $k = 20$ (f) $k = 40$

Figure 3.2: Latent variable recovery results for the latent causal graph shown in orange and triangle causal graph shown in blue, where $\text{card}(X) = m, \text{card}(Y) = n, \text{card}(Z) = k$. For $n = m = 20$, entropy of the latent variable recovered by *LatentSearch* that makes X and Y conditionally independent given the latent is shown against the entropy of the true latent. In (a) and (b), points above the red line are samples for which there is no latent that makes X, Y conditionally independent. As observed, there is a region of low-entropy latent variable regime for which *LatentSearch* can be used to distinguish between the two causal graphs with a properly chosen threshold, e.g., if the entropy of the true latent is less than 3 bits for $|X| = |Y| = 20$.

causal graph: In this experiment, we assume we know the true upper bound on the maximum entropy of the latent variable and used this as the threshold for *InferGraph* (Algorithm 3). Figure 3.2d shows that when latent variable has cardinality less than $\min\{m, n\}$, we cannot find a latent that makes them (almost) conditionally independent. Therefore the *InferGraph* algorithm has almost perfect accuracy. In Figure 3.2e, cardinality of the latent is chosen to



(a) (b) (c)

Figure 3.3: Performance of Causal Inference algorithm, when latent entropy is not known for $m = n = k$ is shown as n is increased. We use different thresholding functions for entropy threshold of the latent. Three thresholds are used $\theta = 2, 0.5\min\{H(X), H(Y)\}$ and $\min\{H(X), H(Y)\} - 1$. *(a)* Average accuracy of causal inference algorithm over all samples, from both *latent graph* and *triangle graph*. As can be seen, thresholding at $\min\{H(X), H(Y)\} - 1$ allows us to distinguish the two causal graph with probability very close to 1, supporting Conjecture 2. *(b)* $\mathbb{P}$(Est.: Latent|True: Latent). *(c)* $\mathbb{P}$(Est.: Complete|True: Complete). All thresholds perfectly classify the complete graphs, whereas latent graps are accurately classified only for the threshold $\theta = \min\{H(X), H(Y)\} - 1$.

be the same as the cardinality of X, Y. As observed, for entropy values less than 3.5 bits, classification is near perfect. When the cardinality of the latent variable is bigger than the cardinality of X, Y (Figure 3.2f), accuracy goes down earlier, around 3 bits. This simulation illustrates that, if we know an upper bound on the entropy of the latent variable which is smaller than $\log(n)$, the *InferGraph* algorithm can be used to detect the true causal graph.

Next, to check the validity of our conjecture, we run *InferGraph*, where we set the threshold θ for the latent entropy as a function of $\min\{H(X), H(Y)\}$. The results are given in Figure 3.3. We observe that as n is increased, accuracy goes to 1 for $\theta = \min\{H(X), H(Y)\} - 1$, whereas accuracy goes to 0 for

$\theta = 2, \theta = 0.5 \min\{H(X), H(Y)\}$. This supports the hypothesis that $a = 1$ in Conjecture 2.

We compared our algorithm with gradient descent and NMF, which we outperform in terms of convergence and performance. We also apply EM algorithm to the output of LatentSearch and observe that EM does not help recover better solutions, in terms of $I(X;Y|Z), H(Z)$ trade-off. These results are given in the supplementary material in Section B.5.

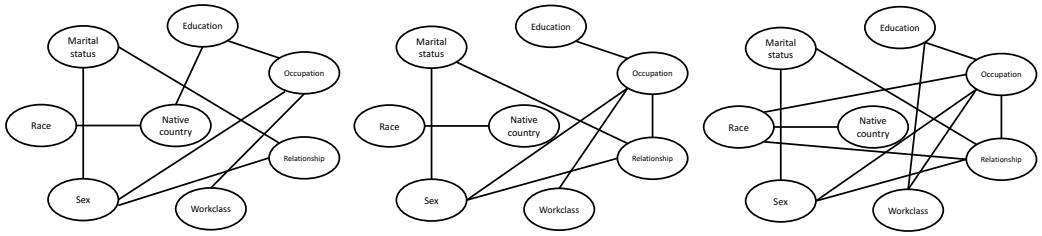
3.5.2 Causal Graph Skeleton Recovery on Adult Dataset

In this section, we study the Adult dataset from UCI Machine Learning Repository [28]. This dataset has 14 attributes 8 of which are discrete: workclass, education, marital status, occupation, relationship, race, sex, native country. It has 45222 samples (after dropping rows with missing values) [5]. The abundance of samples and small number of discrete states for these variables allows us to accurately estimate the joint probability distributions.

Our objective is to test the detection accuracy of *InferGraph* on a real causal graph. Since we are not given information about the latents on this graph, we assume there are no latents. We initialize with the complete undirected graph on the graph vertices. We apply our algorithm on every pair of observed variables X, Y and estimate the joint distribution $p(x, y)$. We then run *LatentSearch* for 100 β values linearly chosen in the range $[0, 0.025]$ for 1000

[5]Data requires mild cleaning: For example before cleanup, "United-states" and "united-states" are treated as different states for the *native-country* variable.

iterations for every β value. Based on the obtained latent variable constructions, we find the minimum entropy the latent variable requires to make the conditional mutual information $I(X;Y|Z)$ less than 0.0005, which we call this $h_{\min}$. For each X, Y, we compare $h_{\min}$ with the threshold $\theta = 0.8 \min\{H(X), H(Y)\}$: If $h_{\min} < \theta$, we remove the edge $X - Y$, else we keep it. After running this algorithm for every pair, we obtain an estimate of the graph skeleton. Our result is shown in Figure 3.4a along with the skeletons obtained by hill-climbing algorithm that uses BIC score, shown in Figure 3.4b, and the skeleton obtained via PC algorithm, shown in Figure 3.4c. These structures are given in [88]. Surprisingly, our results are identical to the one recovered by hill-climbing algorithm except for a single edge. However, it should be noted that our algorithm is sensitive to the choices of both thresholds for $I(X;Y|Z)$ and entropy threshold θ. For example, if we set $\theta = \min\{H(X), H(Y)\}$, we recover empty graph, whereas at least some of the edges are excepted to be direct from the context of the dataset. Setting this threshold based on a dataset is an interesting direction, which is left for future work.



(a) (b) (c)

Figure 3.4: Causal graph skeleton recovered by (a) our algorithm (InferGraph) for entropy threshold θ set to $\theta = 0.8 \min\{H(X), H(Y)\}$, (b) by hill-climbing algorithm with BIC score, (c) PC algorithm [88]. InferGraph can recover almost the same graph recovered by the hill-climbing algorithm with BIC score for this thresholding function for θ.

Chapter 4

Cost-optimal Learning of Causal Graphs

In this chapter, we study the following problem: We want to learn a causal graph where each variable has a cost. For each intervention set, the cost is the sum of the costs of all the variables in the set. Total cost is the sum of the costs of the performed interventions. We would like to learn a causal graph with the minimum possible total cost.

4.1 Our Contributions

Cost-optimal learning of causal graphs is a natural problem that, to the best of our knowledge, has not been previously studied except for some special cases as we explain in the related work section. Our results are as follows:

- We show that the problem of designing the minimum cost interventions to learn a causal graph can be solved in polynomial time.
- We study the minimum cost intervention design problem when the number of interventions is limited. We formulate the cost-optimum intervention

This chapter is based on the material from the publication [68]: M. Kocaoglu, Alex Dimakis, Sriram Vishwanath, *"Cost-Optimal Learning of Causal Graphs,"* Proceedings of the 34th International Conference on Machine Learning (ICML), Sydney, Australia, PMLR 70, 2017. The author of this dissertation contributed to the conception of the research problem, theoretical developments and experimental validation.

design problem as an integer linear program. This formulation allows us to identify two causal graph families for which the problem can be solved in polynomial time.

- For general graphs, we develop an efficient greedy algorithm. We also propose an improved variant of this algorithm, which runs in polynomial time when the skeleton of the causal graph is an interval graph.

Our machinery is graph theoretic. We rely on the connection between graph separating systems and proper colorings. Although this connection was previously discovered, it does not seem to be widely known in the literature.

4.2 Background and Notation

In this section, we present a brief overview of Pearl's causality framework and illustrate how interventions are useful in identifying causal relations. We also present the requisite graph theory background. Finally, we explain separating systems: Separating systems are the central mathematical objects for non-adaptive intervention design.

4.2.1 Causal Graphs, Interventions and Learning

A causal graph is a directed acyclic graph (DAG), where each vertex represents a random variable of the causal system. Consider a set of random variables V. A directed acyclic graph D on the vertex set V and edge set E, $D = (V, E)$, is a causal graph if the arrows in the edge set E encode direct causal relations between the variables: A directed edge $X \to Y$ represents

a direct causal relation between X and Y. X is said to be a direct cause of Y. In the structural causal modeling framework [105], every variable X can be written as a deterministic function of its parent set in the causal graph D and some unobserved random variable E_X. E_X is called an exogenous variable and it is statistically independent from the non-descendants of X. Thus $X = f(Pa_X, E_X)$ where Pa_X is the set of the parents of X in D and f is some deterministic function. We assume that the graph is acyclic[1] (DAG) and all the variables except the exogenous variables are observable (causal sufficiency).

The functional relations between the observed variables and the exogenous variables induce a joint probability distribution over the observed variables. It can be shown that the underlying causal graph D is a valid Bayesian network for the joint distribution induced over the observed variables by the causal model. To identify the causal graph, we can check the conditional independence relations between the observed variables. Under the faithfulness assumption [126], every conditional independence relation is equivalent to a graphical criterion called the *d-separation* [2].

In general, there is no unique Bayesian network that corresponds to a given joint distribution: There exists multiple Bayesian networks for a given

[1]Treatment of cyclic graphs require mechanics different than independent exogenous variables, or a time varying system, and is out of the scope of this chapter.

[2]The set of unfaithful distributions are shown to have measure 0. This makes faithfulness a widely employed assumption, even though it was recently shown that almost faithful distibutions may have significant measure [134].

set of conditional independence relations. Thus, it is not possible to uniquely identify the underlying causal graph using only these tests in general. However, conditional independence tests allow us to identify a certain induced subgraph: *Immoralities*, i.e., induced subgraphs on three nodes of the form $X \to Z \leftarrow Y$. An undirected graph G is called the skeleton of a causal directed graph D, if every edge of G corresponds to a directed edge of D, and every non-edge of G corresponds to a non-edge of D. PC algorithm [126] and its variants use conditional independence tests: They first identify the graph skeleton, and then determine all the immoralities. The runtime is polynomial if the underlying graph has constant vertex degree.

The set of invariant causal edges are not only those that belong to an immorality. For example, one can identify additional causal edges based on the fact that the graph is acyclic. Meek developed a complete set of rules in [90, 92] to identify every invariant edge direction, given a set of causal edges and the skeleton. Meek rules can be iteratively applied to the output of the PC algorithm to identify every invariant arrow. The graph that contains every invariant causal arrow as a directed edge, and the others as undirected edges is called the essential graph of D. Essential graphs are shown to contain undirected components which are always *chordal* [3][126, 49] .

Performing experiments is the most definitive way to learn the causal direction between variables. Randomized clinical trials, which aim to measure

[3]A graph is chordal if its every cycle of length 4 or more contains a chord.

the causal effect of a drug are examples of such experiments. In Pearl's causality framework, an experiment is captured through the *do* operator: The *do* operator refers to the process of assigning a particular value to a set of variables. An *intervention* is an experiment where the scientist collects data after performing the *do* operation on a subset of variables. This process is fundamentally different from conditioning, and requires scientist to have the power of changing the underlying causal system: For example, by forcing a patient not to smoke, the scientist removes the causal effect of the patient's urge to smoke which may be caused by a gene. An intervention is called perfect if it does not change any other mechanism of the causal system and only assigns the desired value to the intervened variable. A stochastic intervention assigns the value of the variable of interest to the realizations of another variable instead of a fixed value. The assigned variable is independent from the other variables in the system. This is represented as $do(X = U)$ for some independent random variable U.

Due to the change of the causal mechanism, an intervention removes the causal arrows from Pa_X to X. This change in the graph skeleton can be detected by checking the conditional independences in the post-interventional distribution: The edges still adjacent to X must have been directing away from X before the experiment. The edges that are missing must have been the parents of X. Thus, an intervention on X enables us to learn the direction of every edge adjacent to X. Similarly, intervening on a set of nodes $S \subseteq V$ concurrently enables us to learn the causal edges across the cut (S, S^c).

Given sufficient data and computation power, we can apply the PC algorithm and Meek rules to identify the essential graph. To discover the rest of the graph we need to use interventions on the undirected components. We assume that we work on a single undirected component after this preprocessing step[4]. Hence, the graphs we consider are chordal without loss of generality, since these components are shown to always be chordal [49]. After each intervention, we also assume that the scientist can apply the PC algorithm and Meek rules to uncover more edges. A set of interventions is said to learn a causal graph given skeleton G, if every causal edge of any causal graph D with skeleton G can be identified through this procedure. A set of m interventions is called an *intervention design* and is shown by $\mathcal{I} = \{I_1, I_2, \ldots, I_m\}$, where $I_i \subset V$ is the set of nodes intervened on in the i^{th} experiment.

An intervention design algorithm is called non-adaptive if the choice of an intervention set does not depend on the outcome of the previous interventions. Yet, we can make use of the Meek rules over the hypothetical outcomes of each experiment. Adaptive algorithms design the next experiment based on the outcome of the previous ones. These algorithms are in general hard to design and analyze. They are also impractical when the scientist needs to design the interventions before the experiment starts, e.g., for parallelized experiments.

In this chapter we are interested in the problem of learning a causal graph given its skeleton where each variable is associated with a cost. The

[4]It is shown that learning additional edges in an undirected component does not help identify edges in another undirected component [49].

objective is to non-adaptively design the set of interventions that minimizes the total interventional cost. We prove that, any set of interventions that can learn every causal graph with a given skeleton needs to be a graph separating system for the skeleton. This is, to the best of our knowledge, the first formal proof of this statement.

4.2.2 Separating systems, Graphs, Colorings

A separating system on a set of elements is a collection of subsets with the following property: For every pair of elements from the set, there exists at least one subset which contains exactly one element from the pair:

Definition 3. *For set $V = [n] \coloneqq \{1, 2, \ldots, n\}$, a collection of subsets $\mathcal{I} = \{I_1, I_2, \ldots I_m\}$, is called a separating system if for every pair $u, v \in V$, $\exists i \in [m]$ such that either $u \in I_i$ and $v \notin I_i$, or $u \notin I_i$ and $v \in I_i$.*

The subset that contains exactly one element from the pair is said to separate the pair. The number of subsets in the separating system is called the size of the separating system. We can represent a separating system with a binary matrix:

Definition 4. *Consider a separating system $\mathcal{I} = \{I_1, I_2, \ldots I_m\}$ for the set $[n]$. A binary matrix $\mathbf{M} \in \{0,1\}^{n \times m}$ is called the separating system matrix for $\mathcal{I}$ if for any element $j \in [n]$, $\mathbf{M}(j, i) = 1$ if $j \in I_i$ and 0 otherwise.*

Thus, each set element has a corresponding row coordinate, and the rows of $\mathbf{M}$ represent the set membership of these elements. Each column of $\mathbf{M}$

is a 0-1 vector that indicates which elements belong to the set corresponding to that column. See Figure 4.1b for two examples. The definition of every pair being separated by some set then translates to every row of M being different.

Given an undirected graph, a graph separating system is a separating system that separates every edge of the graph.

Definition 5. *Given an undirected graph $G = ([n], E)$, a set of subsets of $[n]$, $\mathcal{I} = \{I_1, I_2, \ldots I_m\}$, is a G-separating system if for every pair $u, v \in [n]$ for which $(u, v) \in E$, $\exists i \in [m]$ such that either $u \in I_i$ and $v \notin I_i$, or $u \notin I_i$ and $v \in I_i$.*

Thus, graph separating systems only need to separate pairs of elements adjacent in the graph. Graph separating systems are considered in [87]. It was shown that the size of the minimum graph separating system is $\lceil \log \chi \rceil$, where χ is the coloring number of G. Based on this, we can trivially extend the definition of separating system matrices to include graph separating systems.

A coloring of an undirected graph is an assignment of a set of labels (colors) to every vertex. A coloring is called proper if every adjacent vertex is assigned a different color. A proper coloring for a graph is optimal if it is the proper coloring that uses the minimum number of colors. The number of colors used by an optimal coloring is the chromatic number of the graph. Optimum coloring is hard to find in general graphs, however it is in P for perfect graphs. Since chordal graphs are perfect, the graphs we are interested in in this chapter can be efficiently colored using minimum number of colors.

For a given undirected graph $G = (V, E)$, the vertex induced subgraph on $S \subset V$ is shown by $G_S = \overline{(S, E)}$.

4.3 Related Work

The framework of learing causal relations from data has been extensively studied under different assumptions on the causal model. *The additive noise assumption* asserts that the effect of the exogenous variables are additive in the structural equations. Under the additional assumptions that the data is Gaussian and that the exogenous variables have equal variances, [106] shows that the causal graph is identifiable. Recently, under the additive linear model with jointly Gaussian variables [107] proposed using the invariance of the causal relations to combine a given set of interventional data.

For the case of two variable causal graphs, there is a rich set of theoretical results for data-driven learning: [54] and [124] show that we can learn a two-variable causal graph under different assumptions on the function or the noise term under the additive noise model. Alternatively, an information geometric appraoch that is based on the *independence of cause and effect* is suggested by [61]. [83] recently proposed using a classifier on the datasets to label each dataset either as X causes Y or Y causes X. The lack of large real causal datasets forced him to generate artificial causal data, which makes this approach dependent on the data generation process. An entropic causal inference framework is recently proposed for the two-variable causal graphs by [69].

The literature on learning causal graphs using interventions without assumptions on the causal model is more limited. For the objective of minimizing the number of experiments, [50] proposes a coloring-based algorithm to construct the optimum set of interventions. [33] introduced the constraint on the number of variables intervened in each experiment. He proved in [34] that, when all causal graphs are considered, the set of interventions to fully identify the causal DAG needs to be a separating system for the set of variables. For example for complete graphs, separating systems are necessary. [56] draws connections between the combinatorics literature and causality via known separating system constructions. [122] illustrates several theoretical findings: They show that the separating systems are necessary even under the constraint that each intervention has size at most k, identify an information theoretic lower bound on the necessary number of experiments, and develop an adaptive algorithm that leverages the Meek rules. To the best of our knowledge, the fact that a graph separating system is necessary for a given causal graph skeleton was unknown until this work. Also, none of these works has an explicit cost function associated with interventions.

4.4 Graph Separating Systems, Colorings, Intervention Design

In this section, we illustrate the relation between graph colorings and graph separating systems, and show how they are useful for non-adaptive intervention design algorithms.

Given a graph separating system $\mathcal{I} = \{I_1, I_2, \ldots, I_m\}$ for the skeleton G of a causal graph, we can construct the set of interventions as follows: For experiment i, intervene on the set of variables in the set I_i. Since $\mathcal{I}$ is a graph separating system, for every edge in the skeleton, there is some i for which I_i intervenes on only one of the variables adjacent to that edge. Since the edge is cut, it can be learned by learning the skeleton of the post-interventional graph, as explained in Section 4.2. Since every edge is cut at least once, an intervention design based on a G-separating system identifies any causal graph with skeleton G.

Graph separating systems provide a structured way of designing interventions that can learn any causal graph. Their necessity however is more subtle: One might suspect that using the Meek rules in between every intervention may eliminate the need for the set of interventions to correspond to a graph separating system. Suppose we designed the first $i-1$ experiments. Applying the Meek rules over all possible outcomes of our first $i-1$ experiments on G may enable us to design the mth experiment in an informed manner, even though we do not get to see the outcome of our experiments. Eventually it might be possible to uncover the whole graph without having to separate every edge. In the following we show that Meek rules are not powerful enough to accomplish this, and we actually need a graph separating system. This fact seems to be known [34, 50], however we could not locate a proof. We provide our own proof:

Theorem 7. *Consider an undirected graph G. A set of interventions $\mathcal{I}$ learns*

every causal graph D with skeleton G if and only if $\mathcal{I}$ is a graph separating system for G.

Proof. See [68]. □

4.4.1 Any Graph Separating System is *Some* Coloring

In this section, we explain the relation between graph separating systems and proper graph colorings. This relation, which is already known [50], is important for us in reformulating the intervention design problem in the later sections.

Let $C : V \to \{0,1\}^m$ be a proper graph coloring for graph G which uses c ($c < 2^m$) colors in total. Colors are labeled by length-m binary vectors. First construct matrix $\mathbf{M}$ as follows: Let i^{th} row of $\mathbf{M}$ be the label corresponding to the color of vertex i, i.e., $C(i)$. Then $\mathbf{M}$ is a G-separating system matrix: Let I_i be the set of row indices of $\mathbf{M}$ for which the corresponding entries in the i^{th} column are 1. Let $\mathcal{I} = \{I_1, I_2, \ldots, I_m\}$ be the set of subsets constructed in this manner from m columns of $\mathbf{M}$. Then $\mathcal{I}$ is a graph separating system for G. To see this, consider any pair of vertices u, v that are adjacent in G: $(u, v) \in E$. Since the coloring is proper, the color labels of these vertices are different, which implies the corresponding rows of $\mathbf{M}$, $\mathbf{M}(u,:)$ and $\mathbf{M}(v,:)$, are different. Hence, there is some column of $\mathbf{M}$ which is 1 in exactly one of the u^{th} and v^{th} rows. Thus, the subset constructed from this column separates the pair of vertices u, v.

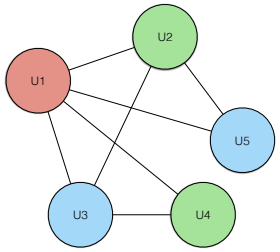
Therefore any proper graph coloring can be used to construct a graph separating system. It turns out that the converse is also true: Any graph separating system can be used to construct a proper graph coloring. This is shown by Cai in [87] within his proof that shows that the minimum size of a graph separating system is $\lceil \log \chi \rceil$, where χ is the chromatic number. We repeat this result for completeness[5]:

Lemma 4 ([87]). *Let $\mathcal{I} = \{I_1, I_2, \ldots, I_m\}$ be a graph separating system for the graph $G = (V, E)$. Let $\mathbf{M}$ be the separating system matrix for $\mathcal{I}$: i^{th} column of $\mathbf{M}$ is the binary vector of length $|V|$ which is 1 in the rows that are contained in I_i. Then the coloring $C(i) = \mathbf{M}(i, :)$ is a proper coloring for G.*

This connection between graph colorings and graph separating systems is important: Ultimately, we want to use graph colorings as a tool for searching over all sets of interventions, and find the one that minimizes a cost function. This is possible due to the characterization in Lemma 4 and the fact that the set of interventions has to correspond to a graph separating system in order to identify any causal graph by Theorem 7.

Along this direction, we have the following simple, yet important observation: We observe that a minimum graph separating system does not have to correspond to an optimum coloring. We illustrate this with a simple example:

[5]Note that this lemma is not formally stated in [87] but rather verbally argued within a proof of another statement.



(a) An undirected graph

	I_1	I_2
U1	1	0
U2	0	1
U3	1	1
U4	0	0
U5	0	0

	I_1	I_2
U1	1	0
U2	0	1
U3	0	0
U4	0	1
U5	0	0

	I_1	I_2
Red	1	0
Green	0	1
Blue	0	0

(b) Graph separating system vs. color separating system

Figure 4.1: (a) An undirected graph with a proper 3 coloring. (b) A graph separating system, which does not separate color classes for any proper coloring of the graph. An example color-separating system is also provided.

Proposition 3. *Consider the undirected graph in Fig. 4.1a. There does not exist any proper 3 coloring of this graph, for which the graph separating system given in Fig. 4.1b separates every node across color classes.*

Proof. Notice that the chromatic number of the given graph is 3. Hence the minimum separating system size is $\lceil \log_2(3) \rceil = 2$. Thus the given graph separating system is a minimum graph separating system. In any proper 3-coloring, $U4$ and $U5$ must have different colors. Hence, any color-separating system separates $U4$ and $U5$. However the rows of the graph separating system which correspond to $U4$ and $U5$ are the same. In other words, any 3-coloring based graph separating system separates $U4$ and $U5$ whereas the graph separating system given in Fig. 4.1a does not. □

This problem can be solved by assigning both vertices $U4$ and $U5$ a new color, hence coloring the graph by $\chi + 1$ colors. We can conclude the following: Suppose we consider the cost-optimum intervention design problem with at

most $\lceil \log(\chi) \rceil$ interventions. When we formulate it as a search problem over the graph colorings, we need to consider the colorings with at most $2^{\lceil \log(\chi) \rceil}$ colors instead of χ colors.

4.5 Cost-Optimal Intervention Design

In this section, we first define the cost-optimal intervention design problem. Later we show that this problem can be solved in polynomial time.

Suppose each variable has an associated cost w_i of being intervened on. We consider a modular cost function: The cost of intervening on a set S of nodes is $w(S) = \sum_{i \in S} w_i$. Our objective is to find the set of interventions with minimum total cost, that can identify any causal graph with the given skeleton: Given the causal graph skeleton G, find the set of interventions $\mathcal{S} = \{S_1, S_2, \ldots, S_m\}$ that can identify any causal graph with the skeleton G, with minimum total cost $\sum_i \sum_{j \in S_i} w_j$. In this section, we do not assume that the number of experiments are bounded and we are only interested in minimizing the total cost. We have the following theorem:

Theorem 8. *Let $G = (V, E)$ be a chordal graph, and $w : V \to \mathbb{R}^+$ be a cost function on its vertices. Let an intervention on set I have cost $\sum_{i \in I} w_i$. Then the optimal set of interventions with minimum total cost, that can learn any causal graph D with skeleton G is given by $\mathcal{I} = \{I_i\}_{i \in [\chi]}$, where I_i is the color class for color i for any χ coloring of the graph $G_{V \setminus S} = \overline{(V \setminus S, E)}$, where S is the maximum weighted independent set of G.*

Proof. See [68]. □

In other words, the optimum strategy is to color the vertex induced subgraph obtained by removing the maximum weighted independent set S and intervening on each color class individually. After coloring the maximum weighted independent set, the remaining graph can always be colored by at most χ colors, i.e., the chromatic number of G. The remaining graph is still chordal. Since optimum coloring and maximum weighted independent set can be found in polynomial time for chordal graphs, $\mathcal{I}$ can be constructed in polynomial time.

4.6 Intervention Design with Bounded Number of Interventions

In this section, we consider the cost-optimum intervention design problem for a given number of experiments. We construct a linear integer program formulation for this problem and identify the conditions under which it can be efficiently solved. As a corollary we show that when the causal graph skeleton is a tree or a clique tree, the cost-optimal intervention design problem can be solved in polynomial time. Later, we present two greedy algorithms for more general graph classes.

To be able to uniquely identify any causal graph, we need a graph separating system by Theorem 7. Hence, we need $m \geq \lceil \log(\chi) \rceil$ since the minimum graph separating system has size $\lceil \log(\chi) \rceil$ due to [87].

4.6.1 Coloring formulation of Cost-Optimum Intervention Design

One common approach to tackle combinatorial optimization problems is to write them as linear integer programs: Often binary variables are used with a linear objective function and a set of linear constraints. The constraints determine the set of feasible points. One can construct a convex object (a convex polytope) based on the set of feasible points by simply taking their convex hull. However this object can not always be described efficiently. If it can, then the linear program over this convex object can be efficiently solved and the result is the optimal solution of the original combinatorial optimization problem. We develop an integer linear program formulation for finding the cost-optimum intervention design using its connection to proper graph colorings.

From Theorem 7, we know that we need the set of interventions to correspond to a graph separating system for the skeleton. From Lemma 4, we know that any graph separating system can be constructed from some proper coloring. Based on these, we have the following key observation: To solve the cost-optimal intervention design problem given a skeleton graph, *it is sufficient to search over all proper colorings*, and find the coloring that gives the graph separating system with the minimum cost. We use the following (standard) coloring formulation: Suppose we are given an undirected graph G with n vertices and t colors are available. Assign a binary variable $x_{i,k} \in \{0, 1\}$ to every vertex-color pair (i, k): $x_{i,k} = 1$ if vertex i is colored with color k, and 0 otherwise. Each vertex is assigned a single color, which can be captured by the equality $\sum_{k\in[t]} x_{i,k} = 1$. Since coloring is proper, every pair

of adjacent vertices are assigned different colors, which can be captured by $x_{i,k} + x_{j,k} \leq 1, \forall (i,j) \in E, \forall k \in [t]$. Based on our linear integer program formulation given in [68], we have the following theorem:

Theorem 9. *Consider the cost-optimal non-adaptive intervention design problem given the skeleton $G = (V, E)$ of the causal graph: Let each node be associated with an intervention cost, and the cost of intervening on a set of variables be the sum of the costs of each variable. Then, the non-adaptive intervention design that can learn any causal graph with the given skeleton in at most m interventions with the minimum total cost can be identified in polynomial time, if the following polytope can be described using polynomially many linear inequalities:*

$$\begin{aligned} \mathcal{C} = \text{conv}\{\mathrm{x} \in \mathbb{R}^{\mathrm{n}\times 2^{\mathrm{m}}} : & \sum\nolimits_{k\in[2^m]} x_{i,k} \leq 1, \forall i \in [n], \\ & x_{i,k} + x_{j,k} \leq 1, \forall (i,j) \in E, \\ & x_{i,k} \in \{0,1\}, \forall i \in [n], k \in [2^m]\}. \end{aligned} \tag{4.1}$$

Proof. See [68]. □

Donne in [31] identifies that when the graph is a tree, one can replace the constraints $x_{i,k} \in \{0,1\}$ with $x_{i,k} \geq 0$ for all $(i,k) \in [n] \times [2^m]$ without changing the polytope in 4.1. He also shows that when the graph is a clique-tree (a graph that can be obtained from a tree by replacing the vertices of the tree with cliques), a simple alternative characterization based on the constraints on the maximum cliques of the graph exists, which can be efficiently described. Based on this and Theorem 9, we have the following corollary:

Corollary 4. *The cost-optimal non-adaptive intervention design problem can be solved in polynomial time if the given skeleton of the causal graph is a tree or a clique tree.*

We can identify another special case for the cost-optimum intervention design problem when the graph is *uniquely colorable*. See [68] for the corresponding result and the details.

4.6.2 Greedy algorithms

In this section, we present two greedy algorithms for the minimum cost intervention design problem for more general graph classes.

Algorithm 4 Greedy Intervention Design for Total Cost Minimization for Chordal Skeleton

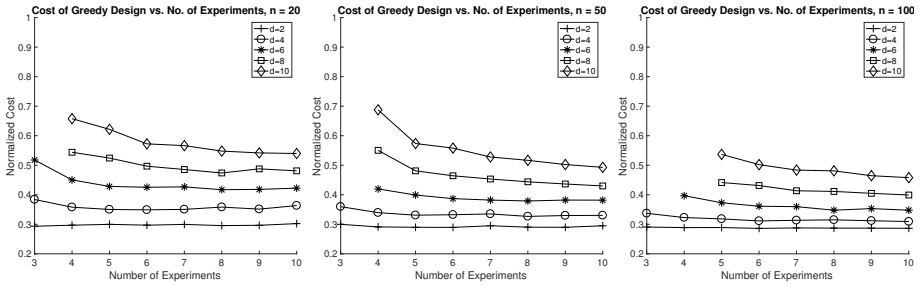
Input: A chordal graph G, maximum number of interventions m, cost w_i assigned to each vertex i.
$r = 2^m, t = 0, G_1 = \overline{(V_1, E)}, V_1 = V$.
$T =$ All binary vectors of length m.
while $r > \chi$ **do**
    Find maximum weighted independent set S_t of G_t.
    Find $u = \arg\min_{x \in T} |x|_1$ (Break ties arbitrarily).
    Assign $M(i,:) =$ to every $i \in S_t$.
    $G_{t+1} = \overline{(V_{t+1}, E)}, V_{t+1} = V_t \backslash S_t$: G_{t+1} is the induced subgraph on the uncolored nodes.
    $r \leftarrow r - 1$, $t \leftarrow t + 1$, $T \leftarrow T - \{u\}$.
end while
Color G_{t-1} with minimum number of colors.
Assign the remaining length-m binary vectors as rows of M to different color classes.
Output: M.

We have the following observation: Consider a coloring $C : V \to [t]$, which uses up to t colors. Consider the graph separating system matrix $\mathbf{M}$

constructed using this coloring, as described in Section 4.4.1. Recall that the i^{th} row of $\mathbf{M}$ is a $\{0,1\}$ vector which represents the label for the color of vertex i, and j^{th} column is the indicator vector for the set of variables included in intervention j. We call the $\{0,1\}$ vector used for color k as the coloring label for color k. The separating property does not depend on the color labels: Using different labels for different colors is sufficient for the graph separating property to hold. However, the number of 1s of a coloring label determines how many times that variable is intervened on using the corresponding intervention design. Hence, we can choose the coloring labels from the binary vectors with small weight, given the choice. Moreover, the column index of a 1 in a certain row does not affect the cost since in a non-adaptive design, every intervention counts towards the total cost (we cannot stop the experiments earlier unlike adaptive algorithms).

Based on this observation, we can try to greedily color the graph as follows: Suppose we are allowed to use up to m interventions. Thus the corresponding graph separating system matrix $\mathbf{M}$ can have up to m columns, which allows up to 2^m distinct coloring labels. We can greedily color the graph by choosing labels with small weight first: Choose the color label with smallest weight from the available labels. Find the maximum weighted independent set of the graph. Assign the coloring label to the rows associated with the vertices in this independent set. Remove the used coloring label from the available labels, update the graph by removing the colored vertices and iterate.

However, this type of greedy coloring could end up using many more



(a) n=20 (b) n=50 (c) n=100

Figure 4.2: Exponential weights $w_i \sim exp(1)$. n: no. of vertices, d: Sparsity parameter of the chordal graph. Each datapoint is the average cost incurred by the greedy intervention design over 1000 randomly sampled causal graphs for a given number of experiments. The expected average cost of all the edges is $\mathbb{E}[w_i] = 1$. The cost incurred by the intervention design is normalized by n. As observed, the cost incurred increases gradually as the number of experiments are reduced, or graph becomes denser. For sparse graphs, proposed construction incurs low cost even for up to 3 experiments.

colors than allowed. Indeed one can show that greedily coloring a chordal graph using maximum independent sets at each step cannot approximate the chromatic number within an additive gap for all graphs. Thus, this vanilla greedy algorithm may use up all 2^m available colors and still have uncolored vertices, even though $\chi < 2^m$. To avoid this, we use the following modified greedy algorithm: For the first $2^m - \chi$ steps, greedily color the graph using maximum weighted independent sets. Use the last χ colors to color the remaining uncolored vertices. Since the graph obtained by removing colored vertices have at most the same chromatic number as the original graph, χ colors are sufficient. The remaining graph is also chordal since removing vertices do not change the chordal property, hence finding a coloring that uses χ colors can be done efficiently. This algorithm is given in Algorithm 4.

We can improve our greedy algorithm when the graph is an interval graph, which is a strict subclass of the chordal graphs. Note that there are $\binom{m}{t}$ binary labels of length m with weight t. When we use these $\binom{m}{t}$ vectors as the coloring labels, the corresponding intervention design requires every variable with these colors to be intervened on exactly t times in total. Then, rather than finding the maximum independent set at iteration t, we can find the maximum weighted $\binom{m}{t}$-colorable subgraph, and use all the coloring labels of weight t. The cost of the colored vertices in the intervention design is t times their total cost. We expect this to create a better coloring in terms of the total cost, since it colors a larger portion of the graph at each step. Finding the maximum weighted k colorable subgraph is hard for non-constant k in chordal graphs, however it can be solved in polynomial time if the graph is an interval graph [141]. For this modified algorithm, see [68]. Notice that when $m >> \log n$, the number of possible coloring labels is super-polynomial in n, which seem to make the algorithms run in super-polynomial time. However, when $m >> \log n$, we can only use the first n color labels with the lowest weight, since a proper coloring on a graph with n vertices can use at most n colors in total.

4.7 Experiments

In this section, we test our greedy algorithm to construct intervention designs over randomly sampled chordal graphs. We follow the sampling scheme proposed by [122] (See [68] for details). The costs of the vertices of the graph

are selected from i.i.d. samples of an exponential random variable with mean 1. The total cost of all variables is then the same as the number of variables n in expectation. We normalize the cost incurred by our algorithm with n and compare this normalized cost for different regimes. The parameter d is a parameter that determines the sparsity of the graph: Graphs with larger d are expected to have more edges. See [68] for the details of how the parameter d affects the probability of an edge. We limit the simulation to at most 10 experiments (x-axis) and observe the effect of changing the number of variables n and parameter d.

Algorithm 4 requires a subroutine that can find the maximum weighted independent set of a given chordal graph. We implement the linear-time algorithm by Frank [39] for finding the maximum weighted independent set of a chordal graph. For the details of Frank's algorithm, see [68].

We observe that the main factor that determines the average incurred cost is sparsity of the graph: The number of edges compared to the number of nodes. For a fixed n, reducing d results in a smaller average cost by increasing the sparsity of the graph. For a fixed d, increasing n reduces the sparsity, which is also shown to reduce the average cost incurred by the greedy intervention design. See [68] for additional simulations where the costs are chosen as the i.i.d. samples from a uniform random variable over the interval $[0, 1]$.

Chapter 5

Experimental Design for Learning Causal Graphs with Latent Variables

In this paper, we consider the problem of learning causal structures with latent variables using interventions. Our objective is not only to learn the causal graph between the observed variables, but also the location of all the unobserved confounding variables. Our approach is stage-wise: We first learn the observable graph, i.e., the induced graph between observable variables. Next we learn the existence and location of latent variables given the observable graph. We first show that a naive approach for learning a sparse observable graph requires $\mathcal{O}(n)$ interventions for a causal graph with n observed variables when latent variables are present. It is rarely feasible to perform many interventions in practice due to costs, technical, or ethical considerations. We propose an efficient randomized algorithm to ameliorate this problem that can learn the observable causal graph using $\mathcal{O}(d \log^2 n)$ interventions where d is the degree of the graph. We further propose an efficient deterministic variant

This chapter is based on the material from the publication [71]: M. Kocaoglu*, K. Shanmugam*, E. Bareinboim, *"Experimental design for learning causal graphs with latent variables,"* Proceedings of the 31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA. The author of this dissertation contributed to the conception of the research problem and the theoretical developments.

which uses $\mathcal{O}(\log n + l)$ interventions, where l is the longest directed path in the graph. This algorithm is useful for different classes of graphs, including common cases such as bipartite, time-series, and relational type of systems. In the next stage, we learn the existence and location of the latent variables between non-adjacent variables. We observe that a naive approach for this task requires $\mathcal{O}(n^2)$ interventions. Instead, we propose an algorithm that uses only $\mathcal{O}(d^2 \log n)$ interventions that can learn the latents between both non-adjacent and adjacent variables. Our algorithm assumes oracle access to an algorithm that outputs a size-$\mathcal{O}(d^2 \log(n))$ independent set cover for the non-edges of a given graph. This graph oracle can be efficiently implemented in practice with a randomized algorithm. Combining the two stages, our randomized algorithm can learn the causal graph with latents using $\mathcal{O}(d \log^2 n + d^2 \log(n))$ interventions.

The endeavour of algorithmically learning causal relations may have started from the independent discovery of the IC [135] and PC algorithms [126], which almost identically, and contrary to previously held beliefs, showed the possibility of recovering these relations from purely observational, non-experimental data. A plethora of methods followed this breakthrough, and now we understand, at least in principle, the limits of what can be inferred from purely observational data [124, 54, 81, 106, 69]. There are a number of assumptions considered about the data-generating model when attempting to unveil the structure underlying the phenomenon under investigation, and we will not attempt to list them exhaustively here. One of the most popular

assumptions, however, is that the data-generating model is *causally sufficient*, which means that no latent (unmeasured) variable affecting more than one observed variable exists. In practice, this is a very stringent condition since the existence of latents affecting more than one observed variable, and generating what is called *confounding bias*, is one of the main concerns of empirical scientists. The problem of causation is deemed challenging in most of the empirical fields because scientists recognize that not all the variables influencing the observed phenomenon can be measured. The general question that arises is then how much of the surface, observed behavior of the system is truly causal, or whether it is due to some external, unobserved forces [105]. Answering this question in a principled way has far-reaching implications for exploring and understanding how nature works, and for better decision-making.

To account for the latent variables, the IC* [135] and FCI [126] algorithms were introduced, which showed the possibility of recovering causal structures *even when* latent variables may be confounding the observed behavior [1]. One of the main challenges faced by this family of algorithms is that although ancestral relations can be learned as well as certain causal edges [142, 17], many observationally equivalent architectures cannot be distinguished. For instance, Fig. 5.1(a),(b) depict two data-generating models that is not distinguishable from passive data, whether A directly causes C or only indirectly (there exists no separator between A and C due to the presence of the latent L). Despite

[1]Hereafter, we refer to a *latent variable* as any variable that is not measured and affects more than one observed variable. In Fig. 5.1(b), note the latent variable L confounding the relationship between B and C.

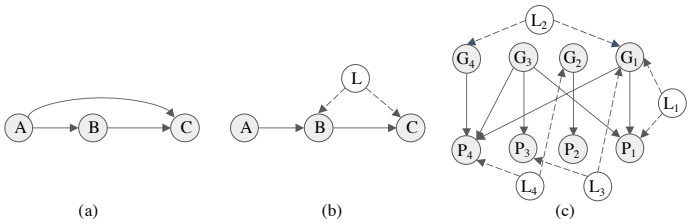


Figure 5.1: (a), (b): Different causal graphs that cannot be distinguished using observational data. (c): A bipartite causal graph between genes and phenotypes with multiple latent variables.

the practical challenges (e.g., finite samples, selection bias, missing data), we now have a complete characterization of what structures are recoverable from observational data [2, 126, 143].

In fact, inferences will be constrained within some type of equivalence class, i.e., a collection of models that are all compatible with the constraints implied over the observational distribution. Some works attempted to use interventional design and interventional data to move from the equivalence class to a specific graph, but almost exclusively considering causally sufficient systems [34, 49, 50, 122]. In [34, 56], the connections between the causal graph learning problem and a separating system, a specific set system, are illustrated. [56] shows that one can design interventions where each intervention contains at most a certain number of variables using the existing separating system constructions. In [49, 50], the authors develop the concept of interventional equivalence class, and using the existing separating system constructions, show that $\lceil \log \chi \rceil$ experiments are necessary and sufficient for learning any causal graph, where χ is the chromatic number of the graph. In [122], authors extend

the results in the regime with bounded sized-interventions.

For causally insufficient systems, there is a plethora of work trying to identify plausible causal structures given interventional data collected a priori under various assumptions[125, 118, 107, 103, 58, 24, 132, 86], but without the goal of designing the optimal interventions. In [125], authors develop algorithms for finding the equivalent causal graphs under latent variables for linear models. Authors in [118] consider removing the effect of latent common causes, under the assumption of additive noise models. Authors in [107] propose using a principle of invariance under different interventions. However, their objective is not to design interventions, but rather identify the plausable causal model, given unknown interventions performed a priori. They also do not consider latent variables, although an example is used to demonstrate the potential extension. Authors in [86] provide an algorithm to construct the casual graph from interventional datasets. The algorithm does not apply to hard interventions. However, it can handle unknown targets for interventions. In [103], authors consider the problem of observational learning of the ancestral relations between observable variables under the presence of latent variables, and propose a dynamic programming algorithm. In the sequence of works [24, 58, 132], authors develop algorithms for finding causal graph upto equivalence classes under latent variables by exploiting available a-priori interventional data. [24] deals with experiments with unknown targets as latents in the system and derives a sound algorithm . [58, 132] provide SAT-based algorithms that encode observed conditional independencies across

data sets as constraints on the graph. [58] can handle cyclic graphs while requiring perfect conditional independency oracles. [132] can handle imperfect oracles but works only on acyclic graphs. Perhaps the most relevant paper to our setup is [93]. Authors identify the experiments necessary to perform in order to identify the causal graph under latent variables, given the output of an observational learning algorithm, such as FCI. However, they are not interested in minimizing the number of experiments.

In this paper, we propose the first algorithm for learning a causal graph with latent variables along with interventional design in a non-parametric regime. It is known that $\log(n)$ interventions are necessary (across all graphs) and sufficient to learn a causal graph without latent variables [50], and we show, perhaps surprisingly, that there exists an algorithm that can learn any causal graph with latent variables, that is not very far from this result. More specifically, our contributions are as follow:

- We introduce a deterministic [2] algorithm that can learn any causal graph and the existence and location of the latent variables using $\mathcal{O}(d\log(n)+l)$ interventions, where d is the largest degree and l is the longest directed path of the causal graph.
- We design a randomized algorithm that can learn the observable graph and all the latent variables using $\mathcal{O}(d\log^2(n)+d^2\log(n))$ interventions with high

[2] We assume access to an oracle which outputs a size-$\mathcal{O}(d^2\log(n))$ independent set cover for the non-edges of a given graph. This oracle can be implemented using another randomized algorithm in practice as we explain in Section 5.4.

probability, where d is the largest degree (in-degree + out-degree).

/[3]/ The first algorithm should be useful in practical settings where the longest directed path is not very deep, e.g., $(\mathcal{O}(\log(n)))$. This includes bipartite, time-series, and relational type of domains where the number of variables is potentially high but the topology is somewhat sparse. As an example application, consider the problem of inferring the causal effect of a set of genes on a set of phenotypes, that could be cast as learning a bipartite causal system (Fig. 5.1(c)). For the more general setting, we introduce a randomized algorithm that with high probability is capable of unveiling the true causal structure.

Background

We will assume for simplicity of exposition that all the random variables involved are discrete throughout this work. We use the language of Structural Causal Models (SCM) [105, pp. 204-207]. Formally, an SCM $\mathcal{M}$ is a 4-tuple $\langle \mathcal{U}, \mathcal{V}, \mathcal{F}, P(u) \rangle$, where $\mathcal{U}$ is a set of exogenous (unobserved, latent) variables, $\mathcal{V}$ is a set of endogenous (measured) variables. The set of exogenous variables are divided into two disjoint classes: Exogenous variables with one observable child, denoted by $\mathcal{E}$, exogenous variables with two observable children, denoted by $\mathcal{L}$. $\mathcal{F}$ represents a collection of functions $\mathcal{F} = \{f_i\}$ such that each endogenous variable $V_i \in \mathcal{V}$ is determined by a function $f_i \in F$: Each f_i is a mapping from the respective domain of the exogenous variables associated with V_i and

[3]Write note contrasting CBN and SCM

a set of observable variables associated with V_i, called PA_i, into V_i. The set of exogenous variables associated with V_i can be divided into two classes, the one with a single observable child, denoted by $\mathcal{E}_i \in \mathcal{E}$, and those with two observable children, denoted by $\mathcal{L}_i \subseteq \mathcal{L}$. Hence f_i maps from the domain of $\mathcal{E}_i \cup PA_i \cup \mathcal{L}_i$ to V_i. The entire set $\mathcal{F}$ forms a mapping from $\mathcal{U}$ to $\mathcal{V}$. The uncertainty is encoded through a product probability distribution over the exogenous variables $P(\mathcal{E}, \mathcal{L})$. For simplicity, with slight overuse of notation, we refer to $\mathcal{L}$ as the set of latent variables, and $\mathcal{E}$ as the set of exogenous variables.

Within the structural semantics, performing an action $X = x$ is represented through the do-operator, $do(X = x)$, which encodes the operation of replacing the original equation of X by the constant x and induces a submodel $\mathcal{M}_x$. When the intervention acts on a set S of variables, the induced submodel is similarly shown by $\mathcal{M}_S$. For a detailed discussion on the properties of structural models, we refer readers to [105, Ch. 7]. Define $D_\ell = (\mathcal{V} \cup \mathcal{L}, E_\ell)$ to be the causal graph with latents. We define the observable graph to be the induced subgraph on $\mathcal{V}$ which is $D = (\mathcal{V}, E)$.

In practice, we will use a stochastic W_i, an independent random variable taking values uniformly at random in the state space of V_i, to implement an intervention in the variable V_i. Interventions can be on a subset S of the observed variables where each observed variable in S is given distinct deterministic or random values. We denote the post-interventional distribution by $P_S(\cdot)$. A conditional independence statement, e.g., *X is independent from Y given $Z \subset \mathcal{V}$ with respect to causal model* $\mathcal{M}_S$, in shown by $(X \perp\!\!\!\perp Y|Z)_{\mathcal{M}_S}$,

or $(X \perp\!\!\!\perp Y|Z)_S$ when the causal model is clear from the context. These conditional independencies are with respect to the post-interventional joint probability distribution $P_S(\cdot)$. In this paper, we assume that an oracle to conditional independence (CI) tests is available.

The *mutilated* or *post-interventional causal graph*, denoted $D_\ell[S] = (\mathcal{V} \cup \mathcal{L}, E_\ell[S])$, is identical to D_ℓ except that all the incoming edges incident on any vertex in the interventional set S is absent, i.e., $E_\ell[S] = E_\ell - \{(Y, V) : V \in S,\ (Y, V) \in E_\ell\}$. We define the *transitive closure*, denoted D_{tc}, of an observable causal DAG D as follows: If there is a directed path from V_i to V_j in D, there is a directed edge from V_i to V_j in D_{tc}. Essentially, a directed edge in D_{tc} represents an ancestral relation in D.

For any directed acyclic graph $D = (V, E)$, a set of nodes $S \subset V$ d-separates two nodes a and b if and only if S blocks all paths between a and b. 'Blocking' is a graphical criterion associated with d-separation. A probability distribution is said to be faithful to a graph, if and only if every conditional independence statement can be read off from the graph using d-separation, see [105] for a review. We assume throughout this paper that faithfulness holds in the observational and post-interventional distributions following the convention in [50].

Results and outline of the paper

The skeleton of the proposed learning algorithms can be split into 3 steps, namely:

$$\emptyset \xrightarrow{\text{(a)}} \text{Transitive Closure, } D_{\text{tc}} \xrightarrow{\text{(b)}} \text{Observable graph, } D = (V, E) \xrightarrow{\text{(c)}} \text{Observable graph with Latent variables, } D_\ell \tag{5.1}$$

Each step requires different tools and graph theoretic concepts:

(a) A pairwise independence test under interventions that reveals ancestral relations. This is combined in an efficient manner with separating systems to discover the transitive closure of D in $\mathcal{O}(\log n)$ interventions.

(b) We rely on the transitive reduction of directed acyclic graphs that can be efficiently computed only from their transitive closure. A key property we observe is that the *transitive reduction reveals a subset of the true edges*. For our randomized algorithm, a sequence of transitive reductions computed from transitive closures (obtained using step (a)) of different post-interventional graphs gives the exact set of true edges.

(c) Given the observational graph, by designing CI tests under suitable interventions, it is possible to discover latent variables between non-adjacent nodes. We use an edge-clique cover algorithm on the complement graph together with these tests to optimize the number of experiments. To discover the latents between adjacent nodes, we introduce a relatively unknown test called the do-see test, i.e., leveraging the equivalence between observing (seeing) and intervening on the node (doing). We again combine this with

induced matching cover of the observable graph to optimize the number of experiments.

The modularity of our approach allows solve various subproblems. For instance, if the observable graph is known a priori, we can learn the latent variables using (c). If the ancestral graph is known, we can apply (b) to discover the observable graph.

5.1 Identifying the Observable Graph: A simple baseline

We discuss a natural and a simple deterministic baseline algorithm that actually finds the observable graph with experiments when confounders are involved. To our knowledge, we do not know of any *provably complete* algorithm that recovers the observable graph under confounding variables and is better than this simple baseline in the worst case. Let us start from the following observation. Suppose $X \to Y$ where X, Y are observable variables and let L be a latent variable such that $L \to X,\ L \to Y$. Consider the post interventional graph $D_\ell[\{X\}]$ where we intervene on X. It is easy to see that, X and Y are dependent in the post interventional graph too because of the direct causal relationship. However, if X is not a direct parent of Y, then in the post interventional graph $D_\ell[\{X\}]$ even with or without the latent L between X and Y, X is independent of Y since X is intervened on.

It is possible to recreate this condition between any target variable X and any one of its direct parents Y when many other observable variables are involved. Simply, we consider the post-interventional graph where we

intervene on all observable variables but X, then in $D_\ell[V - \{X\}]$, Y and X are dependent if and only if $Y \to X$ is a directed edge in the observable graph D, because every other variable except X becomes independent of all other variables in the system in the post interventional graph. Therefore, one needs n interventions, each of size $n-1$ to find out the parent set of every node. We basically show in the next two sections that when the graph D has constant degree, it is enough to do $O(\log^2(n))$ interventions representing the first provably exponential improvement over this baseline for sparse graphs.

5.2 Learning Ancestral Relations

In this section, we show that combinatorial constructions called *separating systems* can be used to construct sequences of pairwise conditional independence tests to discover the transitive closure of the observable causal graph, i.e., the graph that captures all ancestral relations between the observable variables. The following lemma relates the statistical dependencies observed in the post-interventional causal graph with the ancestral relations in the causal graph with latent variables.

Lemma 5. *[Pairwise Conditional Independence Test] Consider a causal graph with latents D_ℓ. Consider an intervention on the set $S \subset \mathcal{V}$ of observable variables. Then, under the post-interventional faithfulness assumption, for any pair $X_i \in S, X_j \in \mathcal{V} \backslash S$, $(X_i \not\perp\!\!\!\perp X_j)_{D_\ell[S]}$ if and only if X_i is an ancestor of X_j in the post-interventional observable graph $D[S]$.*

(a) (b)

Figure 5.2: (a): Illustration of Lemma 1: Consider an intervention on V_4. In the post-interventional distribution, V_4 is dependent with only V_5, V_6, V_7, its descendants, despite latent connections. (b): Transitive closure graph obtained using Algorithm 5. Edges represent ancestral relations, not direct causal connections.

Lemma 5 constitutes, for any ordered pair of variables (X_i, X_j) in the observable graph D, a test for whether X_i is an ancestor of X_j or not. Figure 5.2(a) illustrates Lemma 5. Consider an intervention on V_4. This intervention disconnects V_4 from its parents, including the latent ones. In the post-interventional distribution, V_4 is d-connected to its descendants V_5, V_6, V_7.

Note that a single test is not sufficient to discover the ancestral relation between a pair (X_i, X_j), e.g., if $X_i \to X_k \to X_j$ and $X_i, X_k \in S, X_j \notin S$, the ancestral relation will not be discovered. This issue can be resolved by using a sequence of interventions guided by a separating system, and later finding the transitive closure of the learned graph, which is explained next.

Separating systems were first defined by [65], and has been subsequently used in the context of experimental design in [36]. A separating system on a

Ground Set \ Separating Sets	S_1	S_2	S_3
V_1	1	0	0
V_2	1	1	0
V_3	0	0	0
V_4	0	1	0
V_5	1	0	1
V_6	1	1	1
V_7	0	0	1

(a)

Ground Set \ Strongly Separating Sets	S_1	S_2	S_3	S_4	S_5	S_6
V_1	1	0	1	0	0	0
V_2	1	1	0	1	0	0
V_3	0	0	0	1	1	1
V_4	0	1	0	0	1	1
V_5	1	0	0	1	1	0
V_6	1	1	0	0	1	0
V_7	0	0	1	0	1	1

(b)

Figure 5.3: (a): A separating system on ground set $\{V_1, V_2, V_3, V_4, V_5, V_6, V_7\}$. Each column is the element-set membership vector of the corresponding set. Notice that for every pair of rows, there is a column which is 1 for one of the rows. (b): A strongly separating system. Notice that, for every pair of rows i, j, there are two columns: In one row i is 1 only, in the other, row j is 1 only.

ground set S is a collection of subsets of S, $\mathcal{S} = \{S_1, S_2 \ldots\}$ such that for every pair $i \in S, j \in S$, there is a set that contains only one of i, j, i.e., $\exists k$ such that $i \in S_k$ or $j \in S_k$. For an example separating system construction, see Figure 5.2(b). We require a stronger notion which is captured by a strongly separating system on a ground set $[n]$:

Definition 6. *An (m, n) strongly separating system is a family of subsets $\{S_1, S_2 \ldots S_m\}$ of the ground set $[n]$ such that for any two pairs of nodes i and j, there is a set S in the family such that $i \in S,\ j \notin S$ and also another set S' such that $i \notin S',\ j \in S'$.*

Figure 5.2(c) shows a strongly separating system construction. Although strongly separating systems require more subsets than separating systems, similar to separating systems, one can construct strongly separating systems using $\mathcal{O}(\log(n))$ subsets, as shown in the following lemma:

Lemma 6. *An (m, n) strong separating system exists on a ground set $[n]$ where $m \leq 2\lceil \log n \rceil$.*

We propose Algorithm 5 to discover the ancestral relations between the observable variables. It uses the subsets of a strongly separating system on the ground set of all observable variables as intervention sets, to assure that the ancestral relation between every ordered pair of observable variables is tested.

Algorithm 5 LearnAncestralRelations- Given access to a conditional independence testing oracle (CI oracle), query access to samples from any post-interventional causal model derived out of $\mathcal{M}$ (with causal graph D_ℓ), outputs all ancestral relationships between observable variables, i.e., D_{tc}

LearnAncestralRelations
Input:$\mathcal{M}$
$E = \emptyset$.
Consider a strongly separating system of size at most $2 \log n$ on the ground set $\mathcal{V}$ - $\{S_1, S_2..S_{2\lceil \log n \rceil}\}$.
for i in $[1 : 2\lceil \log n \rceil]$ **do**
    Intervene on the set S_i of nodes.
    for $X \in S_i,\ Y \notin S_i,\ Y \in \mathcal{V}$ **do**
        Use samples from $\mathcal{M}_{S_i}$ and use the CI-oracle to test the following.
        if $(X \not\perp\!\!\!\perp Y)_{D_\ell[S]}$ **then**
            $E \leftarrow E \cup (X, Y)$.
        end if
    end for
end for
return The transitive closure of the graph $(\mathcal{V}, E)$

The following theorem shows the number of experiments and the soundness of Algorithm 5.

Theorem 10. *Algorithm 5 requires only $2\lceil \log n \rceil$ interventions and conditional independence tests on samples obtained from each post-interventional*

distribution and outputs the transitive closure D_{tc}.

5.3 Learning the Observable Graph

We introduce in this section a deterministic and a randomized algorithm for learning the observable causal graph D from ancestral relations. Recall that D is the induced subgraph on the set of observable nodes $\mathcal{V}$, which encodes every direct causal connection between any pair of observable nodes.

5.3.1 A Deterministic Algorithm

Based on the results of Section 5.2, assume that we are given the transitive closure of the observable graph, i.e., D_{tc}. We first have the following causal characterization.

Lemma 7. *For a variable* X_i*, consider an intervention on a set* S *where* $Pa_i \subset S$*. Then* $\{X_j \in S : (X_i \not\perp\!\!\!\perp X_j)_{D[S]}\} = Pa_i$.

Lemma 7 shows that, when the intervention set contains all the parents of a variable X_i, the only variables still dependent with X_i in the post-interventional observable graph are the parents of X_i in the observable graph.

Let the longest directed path of D_{tc} be r (length of the partial order). Consider the partial order $<_{D_{\text{tc}}}$ implied by D_{tc} on the vertex set $\mathcal{V}$. Define $\{T_i : i \in [r+1]\}$ as the unique partitioning of vertices of D_{tc} where $T_i <_{D_{\text{tc}}} T_j, \forall i < j$ and each node in T_i is a set of mutually incomparable elements. In other words, T_i are the set of nodes at layer i of the transitive closure graph

D_{tc}. Define $\mathcal{T}_i = \cup_{k=1}^{i-1} T_k$. We have the following simple observation: $Pa_i \subset \mathcal{T}_i$. This paves the way for Algorithm 6 that leverages Lemma 7.

Algorithm 6 LearnObservableGraph/Deterministic Version - Given the ancestral graph, access to a conditional independence testing oracle (CI oracle) and outputs the graph induced on observable nodes.

LearnObservableGraph/Deteriministic
Input:$\mathcal{M}$
$E = \emptyset$.
for i in $\{r+1, r, r-1, \ldots, 2\}$ **do**
  Intervene on the set $\mathcal{T}_i$ of nodes.
  Use samples from $\mathcal{M}_{\mathcal{T}_i}$ and use the CI-oracle to test the following.
  for X in T_i **do**
    if $(X \not\perp\!\!\!\perp Y)_{D_\ell[\mathcal{T}_i]}$ **then**
      $E \leftarrow E \cup (X, Y)$.
    end if
  end for
end for
return Observable graph

The correctness of Algorithm 6 follows from Lemma 7, which is stated explicitly in the sequel.

Theorem 11. *Let r be the length of the longest directed path in the causal graph D_ℓ. Algorithm 6 requires only r interventions and conditional independence tests on samples obtained from each one of the post-interventional distributions and outputs the observable graph D.*

5.3.2 A Randomized Algorithm

We propose a randomized algorithm that repeatedly uses the ancestor graph learning algorithm from the previous section to learn the observable

graph [4]. One of the key structures that we compute in our algorithm is the transitive reduction of a directed graph:

Definition 7 (Transitive Reduction)**.** *Given a directed acyclic graph* $D = (V, E)$*, let its transitive closure be* D_{tc}*. Then* $\text{Tr}(D) = (V, E_r)$ *is a directed acyclic graph with minimum number of edges such that its transitive closure is identical to* D_{tc}*.*

Lemma 8. *[1]* $\text{Tr}(D)$ *is known to be unique whenever* D *is acyclic. Further, the set of directed edges of* $\text{Tr}(D)$ *is a subset of directed edges of* D*, i.e.,* $E_r \subset E$*. Computing transitive reduction of a directed acyclic graph* D *takes exactly the same time as transitive closure of a directed acyclic graph* D *which takes time* $\text{poly}(n)$*.*

We note that $\text{Tr}(D) = \text{Tr}(D_{\text{tc}})$. Now, we provide an algorithm that outputs an observable graph based on samples from the post-interventional distribution after a sequence of interventions. Let us assume an ordering π on the observable vertices $\mathcal{V}$ that satisfies the partial order relationships in the observable causal graph D. The key insight behind the algorithm is given by the following Lemma.

Lemma 9. *Consider an intervention on a set* $S \subset \mathcal{V}$ *of nodes in the observable causal graph* D*. Consider the post-interventional observable causal graph* $D[S]$*. Suppose for a specific observable node* V_i*,* $V_i \in S^c$*. Let* Y *be a direct parent*

[4]Note that this algorithm does not require learning the ancestral graph first.

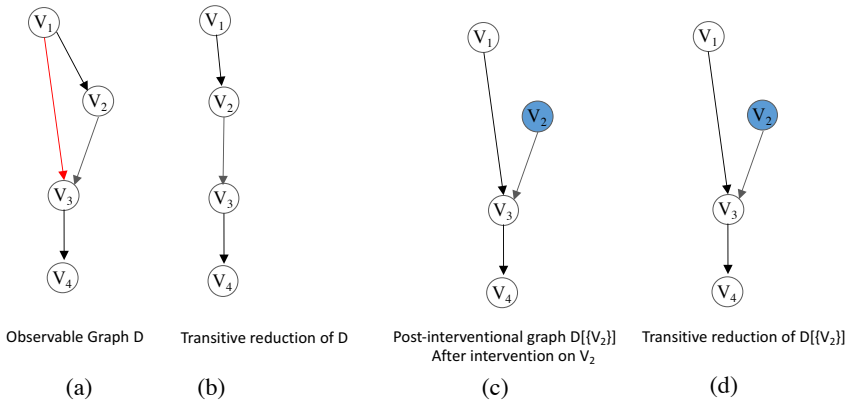


Figure 5.4: Illustration of Lemma 9 - (a) An example of an observable graph D without latents (b): Transitive reduction of D. The highlighted red edge (V_1, V_3) has not been revealed under the operation of transitive reduction. c) Intervention on node V_2 and its post interventional graph $D[\{V_2\}]$ d) Since all parents of V_3 above V_1 in the partial order have been intervened on, by Lemma 9, the edge (V_1, V_3) is revealed in the transitive reduction of $D[\{V_2\}]$.

of V_i in D such that all the direct parents of V_i above Y in the partial order[5] $\pi(\cdot)$ *is in S, i.e., $\{X : \pi(X) > \pi(Y),\ (X, V) \in D\} \subseteq S$. Then,* $\mathrm{Tr}(D[S])$ *will contain the directed edge (Y, V_i) and it can be computed from* $\mathrm{Tr}((D[S])_{\mathrm{tc}})$

The above Lemma is illustrated in Fig. 5.4. The red edge in Fig. 5.4(a) is not revealed in the transitive reduction. The edge is revealed when computing the transitive reduction of the post-interventional graph $D[\{V_2\}]$. This is possible be cause all parents of V_3 above V_1 in the partial order (in this case node V_2) have been intervened on. The above Lemma motivates

[5]The nodes above with respect to the partial order of a graph are those that are closer to the source nodes.

the following algorithm. The basic idea is to intervene in a random fashion, then compute the transitive closure of the post-interventional graph using the algorithm in the previous section, compute the transitive reduction, and then accumulate all the edges found in the transitive reduction at every stage. We will show in Theorem 12 that with high probability, the observable graph can be recovered.

Algorithm 7 LearnObservable- Given access to a conditional independence testing oracle (CI oracle), a parameter $d_{\max}$ outputs induced subgraph between observable variables, i.e. D

LearnObservable/Randomized
Input:$\mathcal{M}, d_{\max}$
$E = \emptyset$.
for i in $[1 : 4cd_{max} \log(n)]$ **do**
  $S = \emptyset$.
  for $V \in \mathcal{V}$ **do**
    $S \leftarrow S \cup V$ randomly with probability $1 - 1/d_{\max}$.
  end for
  $\hat{D}_S = \text{LearnAncestralRelations}(\mathcal{M})$. Let $\hat{D} = (\mathcal{V}, \hat{E})$.
  Compute the transitive reduction of $\hat{D}(\text{Tr}(\hat{D}_S))$ according to the algorithm in [1].
  Add the edges of the transitive reduction to the set E if not already there, i.e. $E \leftarrow E \cup \hat{E}$.
end for
return The directed graph $(\mathcal{V}, E)$.

Theorem 12. *Let the parameter $d_{\max}$ be greater than the maximum in-degree in the observable graph D. Algorithm 7 requires at most $8cd_{\max}(\log n)^2$ interventions and conditional independence tests on samples obtained from each one of the post-interventional distributions, and outputs the observable graph D with probability at least $1 - \frac{1}{n^{c-2}}$.*

Remark. The above algorithm takes as input a parameter $d_{\max}$ that needs to be estimated. One practical option is to gradually increase $d_{\max}$ and run Algorithm 7.

5.4 Learning Latents from the Observable Graph

The final stage of our framework is learning the existence and location of latent variables given the observable graph. We divide this problem into two steps – first, we devise an algorithm that can learn the latent variables between any two variables that are non-adjacent in the observable graph; later, we design an algorithm that learns the latent variables between every pair of adjacent variables.

5.4.1 Baseline Algorithm for Detecting Latents between Non-edges

Consider two variables X and Y such that $X \leftarrow L \rightarrow Y$ and where L is a latent variable. Clearly, to distinguish it from the case where X and Y are disconnected and have no latents, one needs check if $X \not\perp\!\!\!\perp Y$ or not. This is a conditional independence test. For any non edge (X, Y) in the observable graph D, when the observable graph D is known, to check for latents between them, when other variables and possible confounders are around, one has to simply intervene on the rest of the $n - 2$ variables and do a independence test between X and Y in the post interventional graph. This requires a distinct intervention for every pair of variables. If the observable graph has maximum degree $d = o(n)$, this requires $\Theta(n^2)$ interventions. We will reduce this to

$O(d^2 \log n)$ interventions which is an exponential improvement for constant degree graphs.

5.4.2 Latents between Non-adjacent Nodes

We start by noting the following fact about causal systems with latent variables:

Theorem 13. *Consider two non-adjacent nodes X_i, X_j. Let S be the union of the parents of X_i, X_j, $S = Pa_i \cup Pa_j$. Consider an intervention on S. Then we have $(X_i \not\perp\!\!\!\perp X_j)_{\mathcal{M}_S}$ if and only if there exists a latent variable $L_{i,j}$ such that $X_j \leftarrow L_{i,j} \rightarrow X_i$. The statement holds under an intervention S such that $Pa_i \cup Pa_j \subset S,\ X_i, X_j \notin S$.*

The above theorem motivates the following approach: For a set of nodes which forms an independent set, an intervention on the union of parents of the nodes of the independent set allows us to learn the latents between any two nodes in the independent set. We leverage this observation using the following lemma on the number of such independent sets needed to cover all non-edges.

Lemma 10. *Consider a directed acyclic graph $D = (V, E)$ with degree (out-degree+in-degree) d. Then there exists a randomized algorithm that returns a family of $m = \mathcal{O}(4e^2(d+1)^2 \log(n))$ independent sets $\mathcal{I} = \{I_1, I_2, \ldots, I_m\}$ that cover all non-edges of D: $\forall i, j$ such that $(X_i, X_j) \notin E$ and $(X_j, X_i) \notin E$, $\exists k \in [m]$ such that $X_i \in I_k$ and $X_j \in I_k$, with probability at least $1 - \frac{1}{n^2}$.*

Note that this is a randomized construction and we are not aware of any deterministic construction. Our deterministic causal learning algorithm requires oracle access to such a famiy of independent sets, whereas our randomized algorithm can directly use this randomized construction.

Algorithm 8 LearnLatentNonEdge- Given access to a conditional independence testing oracle (CI oracle), observable graph D with max degree d (in-degree+out-degree), outputs all latents between non-edges

LearnLatentNonEdge
Input:$\mathcal{M}, d_{\max}$
$L = \emptyset$.
Apply the randomized algorithm in Lemma 10 to find a family of independent sets $\mathcal{I} = \{I_1, I_2, \ldots, I_m\}$ that cover all non-edges in D such that $m \leq \mathcal{O}(d^2 \log(n))$.
for $j \in [1 : m]$ **do**
  Intervene on the parent set of the nodes in I_j.
  for every pair of nodes X, Y in I_j **do**
    if $(X \not\perp\!\!\!\perp Y)_{D_\ell[I_j]}$ **then**
      $L \leftarrow L \cup \{X, Y\}$.
    end if
  end for
end for
return The set of non-edges L.

Now, we use this observation to construct a procedure to identify latents between non-edges (see Algorithm 8). The following theorem about its performance follows from Lemma 10 and Theorem 13.

Theorem 14. *Algorithm 8 outputs a list of non-edges L that have latent variables between them, given the observable graph D, with probability at least $1 - \frac{1}{n^2}$. The algorithm requires $4e^2(d+1)^2 \log(n)$ interventions where d is the max-degree (in-degree+out-degree) of the observable graph.*

5.4.3 Latents between Adjacent Nodes

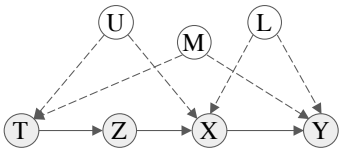
Next, we construct an algorithm that can learn latent variables between the variables adjacent in the observable graph. The first observation is that, the approach of testing (conditional) independence in the post-interventional graph is not helpful. Consider the variables $X \to Y$. To see the effect of the latent path, one needs to cut the direct edge from X to Y. This requires intervening on Y. However, such an intervention disconnects Y from its latent parent. Thus we resort to a different approach compared to the previous stages and exploit a different characterization of causal Bayesian networks called a 'do-see' test.

A do-see test can be described as follows: Consider again the simple graph where $X \to Y$. If there are no latents, we have $\mathbb{P}(Y|X) = \mathbb{P}(Y|\text{do}(X))$. Assume that there is a latent variable Z which causes both X and Y, then excepting the pathological cases[6], $\mathbb{P}(Y|X) \neq \mathbb{P}(Y|\text{do}(X))$. We have the following theorem, which shows that we can perform the do-see test between X, Y under $\text{do}(Pa_X, Pa_Y)$:

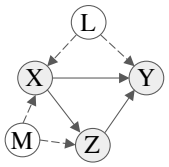
Theorem 15. *[Interventional Do-see test] Consider a causal graph D on the set of observable variables $\mathcal{V} = \{V_i\}_{i\in[n]}$ and latent variables $L = \{L_i\}_{i\in[m]}$ with edge set E. Suppose $(V_i, V_j) \in E$. Then*

$$\Pr(V_j|V_i = v_i, do(Pa_i = pa_i, Pa_j = pa_j)) \\ = \Pr(V_j|do(V_i = v_i, Pa_i = pa_i, Pa_j = pa_j)),$$

[6]These cases will be fully identified later on.



G1: do(PaX) is needed



G2: do(PaY) is needed

Figure 5.5: Left: A graph where intervention on the parents of X is needed for do-see test to succeed. Right: A graph where intervention on the parents of Y is needed for do-see test to succeed.

iff $\nexists k$ *such that* $(L_k, V_i) \in E$ *and* $(L_k, V_j) \in E$*, where* Pa_i *is the set of parents of* V_i *in* V*. Quantities on both sides are invariant irrespective of additional interventions elsewhere.*

Figure 5.5 illustrates the challenges associated with a do-see test in bigger graphs with latents. Graphs $G1$ and $G2$ are examples where parents of both nodes involved in the test need to be included in the intervention set for the Do-see test to work. In $G1$, suppose we condition on X, as required by the 'see' test. This opens up a non-blocking path $X - U - T - M - Y$. Since $X \to Y$ is not the only d-connecting path, it is not necessarily true that $\mathbb{P}(Y|X) = \mathbb{P}(Y|\text{do}(X))$. Now suppose we perform the do-see test under the intervention $\text{do}(Z)$. Then the aforementioned path is closed since X is not a descendant of T in the post interventional graph. Hence we have $\mathbb{P}(Y|X, \text{do}(Z)) = \mathbb{P}(Y|\text{do}(X, Z))$. Similarly $G2$ shows that intervening on the parent set of Y is also necessary.

Next we need a subgraph structure to perform multiple do-see tests at once in order to efficiently discover the latents between the adjacent nodes.

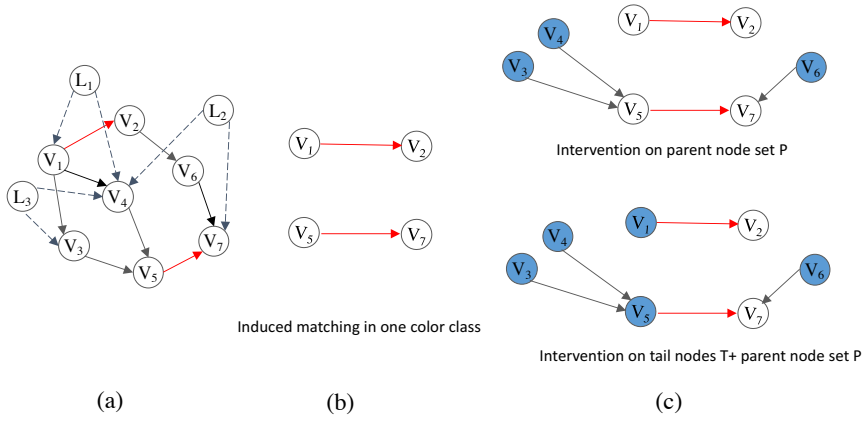


Figure 5.6: Illustration of Theorem 7 and Algorithm 9 - (a) An example of an observable graph with latents with an induced matching highlighted (b): An induced matching (color class) under consideration in the outer loop of Algorithm 9. c) For every color class, only two interventions are needed. One intervenes on the observable parents of all nodes in the color class present outside it, i.e. nodes V_3, V_4 and V_6. The second intervention intervenes on the parent set along with the tail nodes in every edge of the color. This is sufficient to carry out all do-see tests in parallel for the color class.

The naive approach of performing the test for every edge would take $\mathcal{O}(n)$ even in graphs with constant degree. We use strong edge coloring of sparse graphs.

Definition 8. *A strong edge coloring of an undirected graph with k colors is a map $\chi : E \to [k]$ such that every color class is an induced matching. Equivalently, it is an edge coloring such that any two nodes adjacent to distinct edges with the same color are non-adjacent.*

It turns out sparse graphs of maximum degree d (in-degree+out-degree) can be strongly edge-colored with at most $2d^2$ colors.

Lemma 11. *[12] A graph of maximum degree d can be strongly edge-colored with at most $2d^2$ colors. A simple greedy algorithm that greedily colors edges in sequence without violating adjacency condition on coloring achieves this.*

Now observe that a color class of the edges forms an induced matching. We show that due to this, the 'do' part (RHS of Theorem 15) of all the do-see tests in a color class can be performed with a single intervention while the 'see' part (RHS of Theorem 15) can be again performed with another intervention. We argue that we need exactly two different interventions per color class. The following theorem uses this property to prove correctness of Algorithm 9. We pictorially illustrate this key idea of using only two post-interventional distributions to carry out all do-see tests of Theorem 15 for all edges in a single color class simultaneously in Fig. 5.6.

Theorem 16. *Algorithm 9 requires at most $4d^2$ interventions and outputs all latents between the edges in the observable graph.*

5.5 Conclusions

Learning cause-and-effect relations is one of the fundamental challenges in science. We studied the problem of learning causal models with latent variables using experimental data. Specifically, we introduced two efficient algorithms capable of learning direct causal relations (instead of ancestral relations) and finding the existence and location of potential latent variables.

Algorithm 9 LearnLatentEdge- Observable graph D with max degree d (in-degree+out-degree), outputs all latents between edges

LearnLatentEdge
Input: $\mathcal{M}, d$
$L = \emptyset$.
Apply the greedy algorithm in Lemma 11 to color the edges of D with $k \leq 2d^2$ colors.
for $j \in [1 : k]$ **do**
  Let A_j be the nodes involved with the edges that form color class j. Let P_j be the union of parents of all nodes in A_j except the nodes in A_j.
  Let the set of head nodes of all edges be H_j.
  Following loop requires the intervention on the set $H_j \cup P_j$, i.e. $\text{do}(\{H_j, P_j\})$.
  for Every directed edge (V_h, V_t) in color class j **do**
    Calculate the conditional probability: $S(V_h, V_t) = P(V_t|\text{do}(H_j, P_j))$ using samples from the post interventional graph.
  end for
  Following loop requires the intervention on the set P_j.
  for Every directed edge (V_h, V_t) in color class j **do**
    Calculate the conditional probability: $S'(V_h, V_t) = P(V_t|V_h, \text{do}(P_j))$ using samples from the post interventional graph.
    if $S'(V_h, V_t) \neq S(V_h, V_t)$ **then**
      $L \leftarrow L \cup (V_h, V_t)$
    end if
  end for
end for
return The set of edges L that have latents between them.

Chapter 6

CausalGAN: Learning Causal Implicit Generative Models with Adversarial Training

We introduce causal implicit generative models (CiGMs): models that allow sampling from not only the true observational but also the true interventional distributions. We show that adversarial training can be used to learn a CiGM, if the generator architecture is structured based on a given causal graph. We consider the application of conditional and interventional sampling of face images with binary feature labels, such as *mustache, young*. We preserve the dependency structure between the labels with a given causal graph. We devise a two-stage procedure for learning a CiGM over the labels and the image. First we train a CiGM over the binary labels using a Wasserstein GAN where the generator neural network is consistent with the causal graph between the labels. Later, we combine this with a conditional GAN to generate images conditioned on the binary labels. We propose two new conditional GAN architectures: CausalGAN and CausalBEGAN. We show that the optimal generator of the

This chapter is based on the material from the publication [73]: M. Kocaoglu*, C. Snyder*, A. G. Dimakis, S. Vishwanath, *"CausalGAN: Learning Causal Implicit Generative Models with Adversarial Training,"* Proceedings of the Sixth International Conference on Learning Representations (ICLR), May 2018. The author of this dissertation contributed to the conception of the research problem, the theoretical developments and experimental validation.

CausalGAN, given the labels, samples from the image distributions conditioned on these labels. The conditional GAN combined with a trained CiGM for the labels is then a CiGM over the labels and the generated image. We show that the proposed architectures can be used to sample from observational and interventional image distributions, even for interventions which do not naturally occur in the dataset.

6.1 Introduction

An implicit generative model ([95]) is a mechanism that can sample from a probability distribution without an explicit parameterization of the likelihood. Generative adversarial networks (GANs) arguably provide one of the most successful ways to train implicit generative models. GANs are neural generative models that can be trained using backpropagation to sample from very high dimensional nonparametric distributions ([42]). A *generator* network models the sampling process through feedforward computation given a noise vector. The generator output is constrained and refined through feedback by a competitive adversary network, called the discriminator, that attempts to distinguish between the generated and real samples. The objective of the generator is to maximize the loss of the discriminator (convince the discriminator that it outputs samples from the real data distribution). GANs have shown tremendous success in generating samples from distributions such as image and video ([136]).

An extension of GANs is to enable sampling from the class conditional

data distributions by feeding class labels to the generator alongside the noise vectors. Various neural network architectures have been proposed for solving this problem ([94, 101, 5]). However, these architectures do not capture the dependence between the labels. Therefore, they do not have a mechanism to sample images *given a subset of the labels*, since they cannot sample the remaining labels. In this chapter, we are interested in extending the previous work on conditional image generation by *i)* capturing the dependence between labels and *ii)* capturing *the causal effect* between labels. We can think of conditional image generation as a causal process: Labels determine the image distribution. The generator is a non-deterministic mapping from labels to images. This is consistent with the causal graph *"Labels cause the Image"*, denoted by $L \to I$, where L is the random vector for labels and I is the image random variable. Using a finer model, we can also include the causal graph between the labels, if available.

As an example, consider the causal graph between *Gender* (G) and *Mustache* (M) labels. The causal relation is clearly *Gender causes Mustache*, denoted by the graph $G \to M$. Conditioning on *Gender = male*, we expect to see males with or without mustaches, based on the fraction of males with mustaches in the population. When we condition on *Mustache = 1*, we expect to sample from males only since the population does not contain females with mustaches. In addition to sampling from conditional distributions, causal models allow us to sample from various different distributions called *interventional distributions*. An intervention is an experiment that fixes the

(a) Top: Intervened on $Bald = 1$. Bottom: Conditioned on $Bald = 1$. $Male \rightarrow Bald$.

(b) Top: Intervened on $Mustache = 1$. Bottom: Conditioned on $Mustache = 1$. $Male \rightarrow Mustache$.

Figure 6.1: Observational and interventional samples from CausalBEGAN. Our architecture can be used to sample not only from the joint distribution (conditioned on a label) but also from the interventional distribution, e.g., under the intervention do($Mustache = 1$). The two distributions are clearly different since $\mathbb{P}(Male = 1|Mustache = 1) = 1$ and $\mathbb{P}(Bald = 1|Male = 0) = 0$ in the data distribution $\mathbb{P}$.

value of a variable in a causal graph. This affects the distributions of the descendants of the intervened variable in the graph. But unlike conditioning, it does not affect the distribution of its ancestors. For the same causal graph, intervening on *Mustache = 1* would not change the distribution of *Gender*. Accordingly, the label combination (*Gender = female, Mustache = 1*) would appear as often as *Gender = female* after the intervention. Please see Figure 6.1 for some of our conditional and interventional samples, which illustrate this concept on the *Bald* and *Mustache* variables.

In this work we propose *causal implicit* generative models (CiGM): mechanisms that can sample not only from the correct joint probability distributions but also from the correct *conditional* and *interventional* probability distributions. Our objective is not to learn the causal graph: we assume that the true causal graph is given to us. We show that when the generator structure inherits its neural connections from the causal graph, GANs can be

used to train causal implicit generative models. We use Wasserstein GAN (WGAN) ([7]) to train a CiGM for binary image labels, as the first step of a two-step procedure for training a CiGM for the images and image labels. For the second step, we propose two novel conditional GANs called CausalGAN and CausalBEGAN. We show that the optimal generator of CausalGAN can sample from the true conditional distributions (see Theorem 17).

We show that combining CausalGAN with a CiGM on the labels yields a CiGM on the labels and the image, which is formalized in Corollary 5 in Section 6.5. Our contributions are as follows:

- We observe that adversarial training can be used after structuring the generator architecture based on the causal graph to train a CiGM. We empirically show that WGAN can be used to learn a CiGM that outputs *essentially discrete*[1] labels, creating a CiGM for binary labels.
- We consider the problem of conditional and interventional sampling of images given a causal graph over binary labels. We propose a two-stage procedure to train a CiGM over the binary labels and the image. As part of this procedure, we propose a novel conditional GAN architecture and loss function. We show that the global optimal generator provably samples from the class conditional distributions.
- We propose a natural but nontrivial extension of BEGAN to accept labels: using the same motivations for margins as in BEGAN ([13]), we arrive at

[1]Each of the generated labels is sharply concentrated around 0 or 1 (Please see Figure E.4a in the Appendix).

a "margin of margins" term. We show empirically that this model, which we call CausalBEGAN, produces high quality images that capture the image labels.

- We evaluate our CiGM training framework on the labeled CelebA data ([80]). We empirically show that CausalGAN and CausalBEGAN can produce label-consistent images *even for label combinations realized under interventions that never occur during training*, e.g., "woman with mustache"[2].

6.2 Related Work

Using a GAN conditioned on the image labels has been proposed before: In [94], authors propose conditional GAN (CGAN): They extend generative adversarial networks to the setting where there is extra information, such as labels. Image labels are given to both the generator and the discriminator. In [101], authors propose ACGAN: Instead of receiving the labels as input, the discriminator is now tasked with estimating the label. In [128], the authors compare the performance of CGAN and ACGAN and propose an extension to the semi-supervised setting. In [21], authors propose a new architecture called InfoGAN, which attempts to maximize a variational lower bound of mutual information between the inputs given to the generator and the image. To the best of our knowledge, the existing conditional GANs do not allow sampling

[2]This observation is not supported by theory since the distribution over the labels is not strictly positive.

from label combinations that do not appear in the dataset ([127]).

BiGAN ([30]) and ALI ([32]) extend the standard GAN framework by also learning a mapping from the image space to a latent space. In CoGAN ([79]) the authors learn a joint distribution over an image and its binary label by enforcing weight sharing between generators and discriminators. SD-GAN ([29]) is a similar architecture which splits the latent space into "Identity" and "Observation" portions. To generate faces of the same person, one can then fix the identity portion of the latent code. If we consider the "Identity" and "Observation" codes to be the labels then SD-GAN can be seen as an extension of BEGAN to labels. This is, to the best of our knowledge, the only extension of BEGAN to accept labels before CausalBEGAN. It is not trivial to extend CoGAN and SD-GAN to more than two labels. Authors in [5] use CGAN of [94] with a one-hot encoded vector that encodes the age interval. A generator conditioned on this one-hot vector can then be used for changing the age attribute of a face image. Another application of generative models is in compressed sensing: Authors in [16] give compressed sensing guarantees for recovering a vector, if the data lies close to the output of a trained generative model.

Using causal principles for deep learning and using deep learning techniques for causal inference has been recently gaining attention. In [85], the authors observe the connection between GAN layers, and structural equation models. Based on this observation, they use CGAN ([94]) to learn the causal direction between two variables from a dataset. In [84], the authors propose

using a neural network in order to discover the causal relation between image class labels based on static images. In [9], authors propose a new regularization for training a neural network, which they call causal regularization, in order to assure that the model is predictive in a causal sense. In a very recent work [14], authors point out the connection of GANs to causal generative models. However they see image as a cause of the neural net weights, and do not use labels. In an independent parallel work, authors in [43] propose using neural networks for learning causal graphs. Similar to us, they also use neural connections to mimic structural equations, but for learning the causal graph.

6.3 Causality Background

In this section, we give a brief introduction to causality. Specifically, we use Pearl's framework ([105]), i.e., structural causal models (SCMs), which uses structural equations and directed acyclic graphs between random variables to represent a causal model.

Consider two random variables X, Y. Within the SCM framework and under the causal sufficiency assumption[3], X *causes* Y means that there exists a function f and some unobserved random variable E, independent from X, such that *the value of* Y *is determined based on the values of* X *and* E *through the function* f, i.e., $Y = f(X, E)$. Unobserved variables are also called *exogenous*. The causal graph that represents this relation is $X \to Y$. In general, a causal

[3]In a causally sufficient system, every unobserved variable affects not more than a single observed variable.

graph is a directed acyclic graph implied by the structural equations: The parents of a node X_i in the causal graph, shown by Pa_i, represent the *causes* of that variable. The causal graph can be constructed from the structural equations as follows: The parents of a variable are those that appear in the structural equation that determines the value of that variable.

Formally, a structural causal model is a tuple $\mathcal{M} = (\mathcal{V}, \mathcal{E}, \mathcal{F}, \mathbb{P}_{\mathcal{E}}(.))$ that contains a set of functions $\mathcal{F} = \{f_1, f_2, \ldots, f_n\}$, a set of random variables $V = \{X_1, X_2, \ldots, X_n\}$, a set of exogenous random variables $\mathcal{E} = \{E_1, E_2, \ldots, E_n\}$, and a product probability distribution over the exogenous variables $\mathbb{P}_{\mathcal{E}}$. The set of observable variables $\mathcal{V}$ has a joint distribution implied by the distribution of $\mathcal{E}$, and the functional relations $\mathcal{F}$. The causal graph D is then the directed acyclic graph on the nodes $\mathcal{V}$, such that a node X_j is a parent of node X_i if and only if X_j is in the domain of f_i, i.e., $X_i = f_i(X_j, S, E_i)$, for some $S \subset V$. See the Appendix for more details.

An *intervention* is an operation that changes the underlying causal mechanism, hence the corresponding causal graph. An intervention on X_i is denoted as $do(X_i = x_i)$. It is different from conditioning on X_i in the following way: An intervention removes the connections of node X_i to its parents, whereas conditioning does not change the causal graph from which data is sampled. The interpretation is that, for example, if we set the value of X_i to 1, then it is no longer determined through the function $f_i(Pa_i, E_i)$. An intervention on a set of nodes is defined similarly. The joint distribution over the variables after an intervention (post-interventional distribution) can be calculated as follows:

Since D is a Bayesian network for the joint distribution, the observational distribution can be factorized as $\mathbb{P}(x_1, x_2, \ldots x_n) = \prod_{i\in[n]} \mathbb{P}(x_i|Pa_i)$, where the nodes in Pa_i are assigned to the corresponding values in $\{x_i\}_{i\in[n]}$. After an intervention on a set of nodes $X_S \coloneqq \{X_i\}_{i\in S}$, i.e., $do(X_S = \mathbf{s})$, the post-interventional distribution is given by $\prod_{i\in[n]\setminus S} \mathbb{P}(x_i|Pa_i^S)$, where Pa_i^S represents the following assignment: $X_j = x_j$ for $X_j \in Pa_i$ if $j \notin S$ and $X_j = \mathbf{s}(j)$ if $j \in S$[4].

In general it is not possible to identify the true causal graph for a set of variables without performing experiments or making additional assumptions. This is because there are multiple causal graphs that allow the same joint probability distribution even for two variables ([126]). This paper does not address the problem of learning the causal graph: We assume that the causal graph is given to us, and we learn a causal model, i.e., the functions comprising the structural equations for some choice of exogenous variables[5]. There is significant prior work on learning causal graphs that could be used before our method ([126, 52, 23, 54, 56, 51, 122, 83, 107, 37, 110, 69, 68]). When the true causal graph is unknown using a Bayesian network that respects the conditional independences in the data allows us to sample from the correct observational distributions. We explore the effect of the used Bayesian network in Section E.10, E.11.

[4]With slight abuse of notation, we use $\mathbf{s}(j)$ to represent the value assigned to variable X_j by the intervention rather than the jth coordinate of $\mathbf{s}$.

[5]Even when the causal graph is given, there will be many different sets of functions and exogenous noise distributions that explain the observed joint distribution for that causal graph. We are learning one such model.

6.4 Causal Implicit Generative Models

Implicit generative models can sample from the data distribution. However they do not provide the functionality to sample from interventional distributions. We propose *causal implicit generative models*, which provide a way to sample from both observational and interventional distributions.

We show that generative adversarial networks can also be used for training causal implicit generative models. Consider the simple causal graph $X \to Z \leftarrow Y$. Under the causal sufficiency assumption, this model can be written as $X = f_X(E_X), Y = f_Y(E_Y), Z = f_Z(X, Y, E_Z)$, where f_X, f_Y, f_Z are some functions and E_X, E_Y, E_Z are jointly independent variables. The following simple observation is useful: *In the GAN training framework, generator neural network connections can be arranged to reflect the causal graph structure.* Please see Figure 6.2b for this architecture. The feedforward neural networks can be used to represent the functions f_X, f_Y, f_Z. The noise terms (N_X, N_Y, N_Z) can be chosen as independent, complying with the condition that (E_X, E_Y, E_Z) are jointly independent. Note that although we do not know the distributions of the exogenous variables, for a rich enough function class, we can use Gaussian distributed variables ([97]) N_X, N_Y, N_Z. Hence this feedforward neural network can be used to represents the causal models with graph $X \to Z \leftarrow Y$.

The following proposition is well known in the causality literature. It shows that given the true causal graph, two causal models that have the same observational distribution have the same interventional distributions for any intervention. $\mathbb{P}_V$ and $\mathbb{Q}_V$ stands for the distributions induced on the set of

variables in V by $\mathbb{P}_{N_1}$ and $\mathbb{Q}_{N_2}$, respectively.

Proposition 4. *Let*

$$\mathcal{M}_1 = (D_1 = (V, E), N_1, \mathcal{F}_1, \mathbb{P}_{N_1}(.)),$$
$$\mathcal{M}_2 = (D_2 = (V, E), N_2, \mathcal{F}_2, \mathbb{Q}_{N_2}(.))$$

be two causal models, where $\mathbb{P}_{N_1}(.), \mathbb{Q}_{N_2}(.)$ are strictly positive densities. If $\mathbb{P}_V(.) = \mathbb{Q}_V(.)$, then $\mathbb{P}_V(.|do(S)) = \mathbb{Q}_V(.|do(S))$

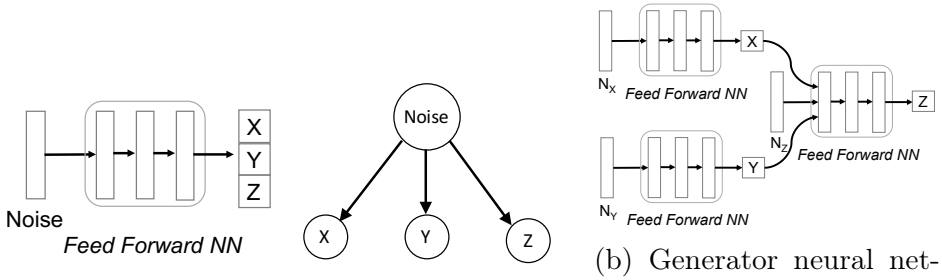
We have the following definition, which ties a feedforward neural network with a causal graph:

Definition 9. *Let $Z = \{Z_1, Z_2, \ldots, Z_m\}$ be a set of mutually independent random variables. A feedforward neural network G that outputs the vector $G(Z) = [G_1(Z), G_2(Z), \ldots, G_n(Z)]$ is called **consistent** with a causal graph $D = ([n], E)$, if $\forall i \in [n]$, $\exists$ a set of feedforward layers f_i such that $G_i(Z)$ can be written as $G_i(Z) = f_i(\{G_j(Z)\}_{j \in Pa_i}, Z_{S_i})$, where Pa_i are the set of parents of i in D, and $Z_{S_i} \coloneqq \{Z_j \ : j \in S_i\}$ are collections of subsets of Z such that $\{S_i : i \in [n]\}$ is a partition of [m].*

Based on the definition, we can define causal implicit generative models as follows:

Definition 10 (CiGM)**.** *A feedforward neural network G with output*

$$G(Z) = [G_1(Z), G_2(Z), \ldots, G_n(Z)], \tag{6.1}$$



(a) Left: Naive feedforward generator. Right: The causal feedforward network represents.

(b) Generator neural network that represents the graph $X \to Z \leftarrow Y$.

Figure 6.2: (a) The causal graph represented by the naive feedforward generator architecture. (b) A neural network implementation of the causal graph $X \to Z \leftarrow Y$: Each feed forward neural net captures the function f in the structural equation model $V = f(Pa_V, E)$.

is called a causal implicit generative model for the causal model $\mathcal{M} = (D = ([n], E), N, \mathcal{F}, \mathbb{P}_N(.))$ *if* G *is consistent with the causal graph* D *and* $\mathbb{P}(G(Z) = \mathbf{x}) = \mathbb{P}_{[n]}(\mathbf{x}) > 0, \forall \mathbf{x}$.

We propose using adversarial training where the generator neural network is consistent with the causal graph according to Definition 9, which is explained in the next section.

6.5 Causal Generative Adversarial Networks

CiGMs can be trained with samples from a joint distribution given the causal graph between the variables. However, for the application of image generation with binary labels, we found it difficult to simultaneously learn the

joint label and image distribution[6]. For this application, we focus on dividing the task of learning a CiGM into two subtasks: First, we train a generative model over the labels, then train a generative model for the images conditioned on the labels. For this training to be consistent with the causal structure, we assume that the image node is always the sink node of the causal graph for image generation problems (Please see Figure E.1 in Appendix). As we show next, our new architecture and loss function (CausalGAN) assures that the optimum generator outputs the label conditioned image distributions, under the assumption that the joint probability distribution over the labels is strictly positive[7]. Then for a strictly positive joint distribution between labels and the image, combining CiGM for only the labels with a label-conditioned image generator gives a CiGM for images and labels (see Corollary 5).

6.5.1 Causal Controller

First we describe the adversarial training of a CiGM for binary labels. This generative model, which we call the *Causal Controller*, will be used for controlling which distribution the images will be sampled from when intervened or conditioned on a set of labels. As in Section 6.4, we structure the Causal Controller network to sequentially produce labels according to the causal graph. Since our theoretical results hold for binary labels, we prefer a generator

[6]Please see the Section E.16 in the Appendix for our primitive result using this naive attempt.

[7]This assumption does not hold in the CelebA dataset: $\mathbb{P}(Male = 0, Mustache = 1) = 0$. However, we will see that the trained model is able to extrapolate to these interventional distributions.

which can sample from an essentially discrete label distribution[8]. However, the standard GAN training is not suited for learning a discrete distribution, since Jensen-Shannon divergence requires the support to be the same for giving meaningful gradients, which is harder with discrete data distributions. To be able to sample from a discrete distribution, we employ WGAN ([7]). We used the model of [48], where the Lipschitz constraint on the gradient is replaced by a penalty term in the loss.

6.5.2 CausalGAN

6.5.2.1 Architecture

As part of the two-step process proposed in Section 6.4 for learning a CiGM over the labels *and* the image variables, we design a new conditional GAN architecture to generate the images based on the labels of the Causal Controller. Unlike previous work, our new architecture and loss function assures that the optimum generator outputs the label conditioned image distributions. We use a pretrained Causal Controller which is not further updated.

Labeler and Anti-Labeler: We have two separate labeler neural networks. *The Labeler* is trained to estimate the labels of images in the dataset. *The Anti-Labeler* is trained to estimate the labels of the images sampled from the generator, where image labels are those produced by the Causal Controller.

Generator: The objective of the generator is 3-fold: producing realistic

[8]Ignoring the theoretical considerations, adding noise to transform the labels artificially into continuous targets also works. However we observed better empirical convergence with this technique.

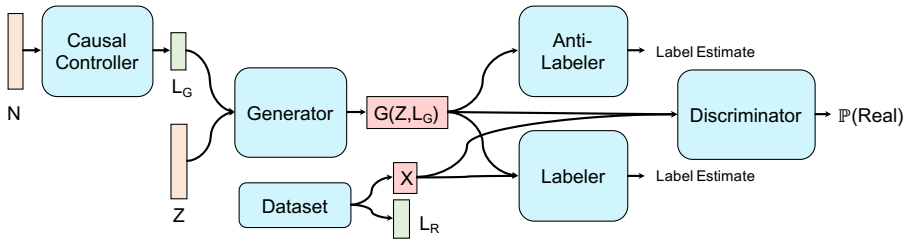


Figure 6.3: CausalGAN architecture: Causal controller is a pretrained causal implicit generative model for the image labels. Labeler is trained on the real data, Anti-Labeler is trained on generated data. Generator minimizes Labeler loss and maximizes Anti-Labeler loss.

images by competing with the discriminator, producing images consistent with the labels by minimizing the Labeler loss and avoiding unrealistic image distributions that are easy to label by maximizing the Anti-Labeler loss.

The most important distinction of CausalGAN with the existing conditional GAN architectures is that it uses an Anti-Labeler network in addition to a Labeler network. Notice that the theoretical guarantee we develop in Section 6.5.2.3 does not hold without the Anti-Labeler. Intuitively, the Anti-Labeler loss discourages the generator network to output only few typical faces for a fixed label combination. This is a phenomenon that we call *label-conditioned mode collapse*. Minibatch-features are one of the most popular techniques used to avoid mode-collapse ([117]). However, the diversity within a batch of images due to different label combinations can make this approach ineffective for combating label-conditioned mode collapse. This effect is most prominent for rare label combinations. In general, using Anti-Labeler helps with faster convergence. Please see Section E.21 in the Appendix for results.

6.5.2.2 Loss Functions

We present the results for a single binary label l. The results can be extended to more labels. For a single binary label l and the image x, we use $\mathbb{P}_r(l, x)$ for the data distribution between the image and the labels. Similarly $\mathbb{P}_g(l, x)$ denotes the joint distribution between the labels given to the generator and the generated images. In our analysis we assume a perfect Causal Controller[9] and use the shorthand $\mathbb{P}_g(l = 1) = \mathbb{P}_r(l = 1) = \rho = 1 - \bar{\rho}$. Let $G(.), D(.), D_{LR}(.)$, and $D_{LG}(.)$ are the mappings due to generator, discriminator, Labeler, and Anti-Labeler respectively.

The generator loss function of CausalGAN contains label loss terms, the GAN loss in [42], and an added loss term due to the discriminator. With the addition of this term to the generator loss, we are able to prove that the optimal generator outputs the class conditional image distribution. This result is also true for multiple binary labels, which is shown in the Appendix.

For a fixed generator, Anti-Labeler solves the following optimization problem:

$$\max_{D_{LG}} \rho \mathbb{E}_{x \sim \mathbb{P}_g(x|l=1)} \left[\log(D_{LG}(x))\right] + \bar{\rho} \mathbb{E}_{x \sim \mathbb{P}_g(x|l=0)} \left[\log(1 - D_{LG}(x)\right]. \tag{6.2}$$

The Labeler solves the following optimization problem:

$$\max_{D_{LR}} \rho \mathbb{E}_{x \sim \mathbb{P}_r(x|l=1)} \left[\log(D_{LR}(x))\right] + \bar{\rho} \mathbb{E}_{x \sim \mathbb{P}_r(x|l=0)} \left[\log(1 - D_{LR}(x)\right]. \tag{6.3}$$

[9]Even for multiple labels, we observe convergence in total variation distance. Please see Figure E.4b.

For a fixed generator, the discriminator solves the following optimization problem:

$$\max_{D} \mathbb{E}_{(l,x)\sim\mathbb{P}_r(l,x)} \left[\log(D(x))\right] + \mathbb{E}_{(l,x)\sim\mathbb{P}_g(l,x)} \left[\log\left(1 - D(x)\right)\right]. \tag{6.4}$$

For a fixed discriminator, Labeler and Anti-Labeler, the generator solves the following problem:

$$\begin{aligned}
&\min_{G} \mathbb{E}_{(l,x)\sim\mathbb{P}_g(l,x)} \left[\log\left(\frac{1 - D(x)}{D(x)}\right)\right] - \rho\mathbb{E}_{x\sim\mathbb{P}_g(x|l=1)} \left[\log(D_{LR}(X))\right] \\
&- \bar{\rho}\mathbb{E}_{x\sim\mathbb{P}_g(x|l=0)} \left[\log(1 - D_{LR}(X))\right] + \rho\mathbb{E}_{x\sim\mathbb{P}_g(x|l=1)} \left[\log(D_{LG}(X))\right] \\
&+ \bar{\rho}\mathbb{E}_{x\sim\mathbb{P}_g(x|l=0)} \left[\log(1 - D_{LG}(X))\right].
\end{aligned} \tag{6.5}$$

6.5.2.3 Theoretical Guarantees

We show that the best CausalGAN generator for the given loss function samples from the class conditional image distribution when Causal Controller samples from the true label distribution and the discriminator and labeler networks always operate at their optimum. We show this result for the case of a single binary label $l \in \{0, 1\}$. The proof can be extended to multiple binary variables, which is given in the Appendix. As far as we are aware of, this is the only conditional generative adversarial network architecture with this guarantee after CGAN[10].

First, we find the optimal discriminator for a fixed generator. Note that in (6.4), the terms that the discriminator can optimize are the same as the GAN

[10]CGAN ([94]) can be shown to have the same guarantee. The difference of our architecture is that we do not feed image labels to the discriminator.

loss in [42]. Hence the optimal discriminator behaves the same. To characterize the optimum discriminator, labeler and anti-labeler, we have Proposition 7, Lemma 22 and Lemma 23 given in the appendix.

Let $C(G)$ be the generator loss for when the discriminator, Labeler and Anti-Labeler are at the optimum. Then the generator that minimizes $C(G)$ samples from the class conditional distributions:

Theorem 17. *Assume $\mathbb{P}_g(l) = \mathbb{P}_r(l)$. Then the global minimum of the virtual training criterion $C(G)$ is achieved if and only if $\mathbb{P}_g(l, x) = \mathbb{P}_r(l, x)$, i.e., if and only if given a label l, generator output $G(z, l)$ has the same distribution as the class conditional image distribution $\mathbb{P}_r(x|l)$.*

Now we can show that our two stage procedure can be used to train a causal implicit generative model for any causal graph where the *Image* variable is a sink node, captured by the following corollary. $\mathcal{L}, \mathcal{I}, \mathcal{Z}_1, \mathcal{Z}_2$ represent the space of labels, images, and noise variables, respectively.

Corollary 5. *Suppose $C : \mathcal{Z}_1 \to \mathcal{L}$ is a causal implicit generative model for the causal graph $D = (\mathcal{V}, E)$ where $\mathcal{V}$ is the set of image labels and the observational joint distribution over these labels are strictly positive. Let $G : \mathcal{L} \times \mathcal{Z}_2 \to \mathcal{I}$ be a generator that can sample from the image distribution conditioned on the given label combination $L \in \mathcal{L}$. Then $G(C(Z_1), Z_2)$ is a causal implicit generative model for the causal graph $D' = (\mathcal{V} \cup \{Image\}, E \cup \{(V_1, Image), (V_2, Image), \ldots (V_n, Image)\})$.*

In Theorem 17 we show that the optimum generator samples from the class conditional distributions given a single binary label. Our objective is to extend this result to the case with d binary labels. First we show that if the Labeler and Anti-Labeler are trained to output 2^d scalars, each interpreted as the posterior probability of a particular label combination given the image, then the minimizer of $C(G)$ samples from the class conditional distributions *given d labels*. This result is shown in Theorem 19 in the appendix. However, when d is large, this architecture may be hard to implement. To resolve this, we propose an alternative architecture, which we implement for our experiments: We extend the single binary label setup and use cross entropy loss terms for each label. This requires Labeler and Anti-Labeler to have only d outputs. However, although we need the generator to capture the joint label posterior given the image, this only assures that the generator captures each label's posterior distribution, i.e., $\mathbb{P}_r(l_i|x) = \mathbb{P}_g(l_i|x)$ (Proposition 8). This, in general, does not guarantee that the class conditional distributions will be true to the data distribution. However, for many joint distributions of practical interest, where *the set of labels are completely determined by the image*[11], we show that this guarantee implies that the joint label posterior will be true to the data distribution, implying that the optimum generator samples from the class conditional distributions. Please see Section E.7 for the formal results and more details.

Remark: Note that the trained causal implicit generative models can

[11]The dataset we are using arguably satisfies this condition.

also be used to sample from the counterfactual distributions if the exogenous noise terms are known. Counterfactual sampling require conditioning on an event and sampling from the push-forward of the posterior distributions of the exogenous noise terms under the interventional causal graph due to a possible intervention. This can be done through rejection sampling to observe the evidence, holding the exogenous noise terms consistent with the observed evidence and interventional sampling afterwards.

6.5.3 CausalBEGAN

In this section, we sketch a simple, but non-trivial extension of BEGAN where we feed image labels to the generator, leaving the details to the Appendix (Section E.8). To accommodate interventional sampling, we again use the Causal Controller to produce labels.

In terms of architecture modifications, we use a Labeler network with a dual purpose: to label real images well and generated images poorly. This network can be seen as both analogous to the original two-roled BEGAN discriminator and analogous to the CausalGAN Labeler and Anti-Labeler.

In terms of margin modifications, we are motivated by the following observations: (1) Just as a better trained BEGAN discriminator creates more useful gradients for image quality, (2) a better trained Labeler is a prerequisite for meaningful gradients for label quality. Finally, (3) label gradients are most informative when the image quality is high. Each observation suggests a margin term; the final observation suggests a (necessary) *margin of margins*

Top: Intervene Mustache=1, Bottom: Condition Mustache=1

Figure 6.4: Intervening/Conditioning on Mustache label in CelebA Causal Graph with CausalGAN. Since $Male \rightarrow Mustache$ in CelebA Causal Graph, we do not expect $do(Mustache = 1)$ to affect the probability of $Male = 1$, i.e., $\mathbb{P}(Male = 1|do(Mustache = 1)) = \mathbb{P}(Male = 1) = 0.42$. Accordingly, the top row shows both males and females with mustaches, even though the generator never sees the label combination $\{Male = 0, Mustache = 1\}$ during training. The bottom row of images sampled from the conditional distribution $\mathbb{P}(.|Mustache = 1)$ shows only male images.

term comparing the first two margins.

6.6 Results

In this section, we train CausalGAN and CausalBEGAN on the CelebA Causal Graph given in Figure E.1. For this, we first trained the Causal Controller (See Section E.11 for Causal Controller results.) on the image labels of CelebA Causal Graph. Please see Section E.19 for implementation details. The results are given in Figures 6.4, 6.5 for CausalGAN and Figures 6.6, 6.7 for CausalBEGAN. The difference between intervening and conditioning is clear through certain features. We implement conditioning through rejection sampling. See [100, 44] for other works on conditioning for implicit generative models.

Top: Intervene Mouth Slightly Open=1, Bottom: Condition Mouth Slightly Open=1

Figure 6.5: Intervening/Conditioning on Mouth Slightly Open label in CelebA Causal Graph with CausalGAN. Since *Smiling* $\rightarrow$ *Mouth Slightly Open* in CelebA Causal Graph, we do not expect *do*(*Mouth Slightly Open*= 1) to affect the probability of *Smiling* = 1, i.e., $\mathbb{P}(Smiling = 1|do(\textit{Mouth Slightly Open}= 1)) = \mathbb{P}(Smiling = 1) = 0.48$. However on the bottom row, conditioning on *Mouth Slightly Open*= 1 increases the proportion of smiling images (From 0.48 to 0.76 in the dataset), although 10 images may not be enough to show this difference statistically.

Top: Intervene Mustache=1, Bottom: Condition Mustache=1

Figure 6.6: Intervening/Conditioning on Mustache label in CelebA Causal Graph with CausalBEGAN. Since $Male \rightarrow Mustache$ in CelebA Causal Graph, we do not expect $do(Mustache = 1)$ to affect the probability of $Male = 1$, i.e., $\mathbb{P}(Male = 1|do(Mustache = 1)) = \mathbb{P}(Male = 1) = 0.42$. Accordingly, the top row shows both males and females with mustaches, even though the generator never sees the label combination $\{Male = 0, Mustache = 1\}$ during training. The bottom row of images sampled from the conditional distribution $\mathbb{P}(.|Mustache = 1)$ shows only male images.

Top: Intervene Narrow Eyes=1, Bottom: Condition Narrow Eyes=1

Figure 6.7: Intervening/Conditioning on Narrow Eyes label in CelebA Causal Graph with CausalBEGAN. Since $Smiling \rightarrow Narrow\ Eyes$ in CelebA Causal Graph, we do not expect $do(Narrow\ Eyes = 1)$ to affect the probability of $Smiling = 1$, i.e., $\mathbb{P}(Smiling = 1|do(Narrow\ Eyes = 1)) = \mathbb{P}(Smiling = 1) = 0.48$. However on the bottom row, conditioning on $Narrow\ Eyes = 1$ increases the proportion of smiling images (From 0.48 to 0.59 in the dataset), although 10 images may not be enough to show this difference statistically. As a rare artifact, in the dark image in the third column the generator appears to rule out the possibility of $Narrow\ Eyes = 0$ instead of demonstrating $Narrow\ Eyes = 1$.

6.7 Conclusion

We proposed a novel generative model with label inputs. In addition to being able to create samples *conditioned* on labels, our generative model can also sample from the *interventional* distributions. Our theoretical analysis provides provable guarantees about correct sampling under such interventions. Causality leads to generative models that are more creative since they can produce samples that are different from their training samples in multiple ways. We have illustrated this point for two models (CausalGAN and CausalBEGAN).

Appendices

Appendix A

Appendix for Entropic Causal Inference

A.1 Proof of Lemma 1

($\Rightarrow$)Assume there exists a causal model $\mathcal{M} = (\{X, Y\}, E, f, X \to Y, p_{X,E})$. Without loss of generality, assume $\mathcal{E} = [m]$ and $p_E = [e_1, e_2, \ldots, e_m]$. We have

$$p(y|x) = \sum_{e \in \mathcal{E}} p(y|x, e)p(e|x) = \sum_{e \in \mathcal{E}} p(y|x, e)p(e) \tag{A.1}$$

since $E \perp\!\!\!\perp X$. Define the matrix $\mathbf{Y}|\mathbf{X_k}(i, j) \coloneqq \mathbb{P}(Y = i|X = j, E = k)$. Then, from (A.1), we can decompose the conditional probability distribution matrix $\mathbf{Y}|\mathbf{X}$ as follows:

$$\mathbf{Y}|\mathbf{X} = \sum_{k \in \mathcal{E}} e_k \mathbf{Y}|\mathbf{X_k}. \tag{A.2}$$

Since f is deterministic, each value of X is mapped to exactly one value of Y, when E is conditioned on. Thus each column of $\mathbf{Y}|\mathbf{X_k}$ has exactly a single 1 with remaining entries being zeros. Thus, each entry of $\mathbf{Y}|\mathbf{X}$ is a subset sum of p_E. Let $S_{i,j}$ represent this subset, i.e., $p(y = i|x = j) = \sum_{k \in S_{i,j}} e_k$. Notice that the xth column of $\mathbf{Y}|\mathbf{X}$ is the conditional distribution $\mathbb{P}(Y|X = x) = \mathbb{P}(f(x, E)|X = x) = \mathbb{P}(f(x, E)) = \mathbb{P}(f_x(E))$, where $f_x(E) \coloneqq f(x, E)$. Since f_x is a deterministic function, each value in its domain maps to exactly one value in its range. This implies that $S_{y,x} = f_x^{-1}(y)$ are disjoint for fixed x,

i.e., $S_{i,j} \neq S_{l,j} \forall l \neq i$. Also, since each value of E must be mapped to a value of Y by f_x, the union of $S_{i,j}$ over i must be the whole support $[m]$.

Define $\mathbf{m}_{i,j}$ to be the length m vector which is 1 in the columns indexed by $S_{i,j}$. Construct $\mathbf{M}$ from the rows $\mathbf{m}_{i,j}$ such that $m_{i,j}$ is the $i+(j-1)n$th row of $\mathbf{M}$. By construction, $\mathbf{M}$ is a block partition matrix. Pick $\mathbf{e} = [e_1, e_2, \ldots, e_m]$. Then we have $vec(\mathbf{Y}|\mathbf{X}) = \mathbf{Me}$.

($\Leftarrow$) For reverse direction, assume there exists matrices $\mathbf{M}, \mathbf{e}$ with block partition $\mathbf{M}$ and $\sum_i e_i = 1$, such that $vec(\mathbf{Y}|\mathbf{X}) = \mathbf{Me}$. Define E to be the random variable independent from X, with probability distribution $p_E \coloneqq \mathbf{e}$ and support $\mathcal{E} = [m]$.

Let $\mathbf{w}_i$ be the ith column of $\mathbf{M}$. Then we have $vec(\mathbf{Y}|\mathbf{X}) = \sum_i e_i \mathbf{w}_i$. Now de-vectorize $\mathbf{Y}|\mathbf{X}$ and $\mathbf{w}_i$ to have

$$\mathbf{Y}|\mathbf{X} = \sum_{i=1}^{m} e_i \mathbf{U}_i, \tag{A.3}$$

where $\mathbf{w}_i = vec(\mathbf{U}_i)$. Each column of $\mathbf{U}_i$, comes from distinct size-n blocks of $\mathbf{M}$ and since $\mathbf{M}$ is block partition, each column of $\mathbf{U}_i$ contains a single 1 $\forall i$. Thus each $\mathbf{U}_i$ represent a valid map $f_i : \mathcal{X} \to \mathcal{Y}$, where $f_i(x)$ is given by the nonzero row of xth column of $\mathbf{U}_i$.

A function f with two input variables X, E, where $E \in \mathcal{E} = [m]$ is completely determined by the set of functions $\{f(X,1), f(X,2), ..., f(X,m)\}$. Let $f(X,e)$ be the function described by the matrix $\mathbf{U}_e$ and f be the function determined by $\{f(X,e), e \in \mathcal{E}\}$. Apply the constructed f on X, E to get

$Z = f(X, E)$. Then we have $\mathbb{P}(Y = y|X = x) = \mathbb{P}(Z = y|X = x)\forall x$, since both Z and Y induce the same conditional distribution $\mathbf{Y}|\mathbf{X}$.

For any set of realizations of $(X, Y) = \{x_i, y_i\}$, we can construct a set of realizations of $E, \{e_i\}$ based on the conditional distribution $\mathbb{P}(E|Z = y_i, X = x_i)$. Then we have $y_i = f(x_i, e_i)$. Also, since $\mathbb{P}(Z = y, X = x) = \mathbb{P}(Y = y, X = x)$ this process induces the same joint distributions between variables X, Y, E and X, Z, E, i.e., $\mathbb{P}(X = x, E = e, Z = y) = \mathbb{P}(X = x, E = e, Y = y)$, and conditional independence statement implied by one holds for the other. Thus $X \perp\!\!\!\perp E$.

A.2 Unidentifiability without assumptions

Here we prove that we can fit causal models in both directions $X \to Y, Y \to X$ given any joint distribution.

Lemma 12. *Let $X \in \mathcal{X}, Y \in \mathcal{Y}$ be discrete random variables with an arbitrary joint distribution $p(x, y)$, where $|\mathcal{X}| = m, |\mathcal{Y}| = n$. Then there exists two causal models $\mathcal{M}_1 = (\{X, Y\}, E, f, X \to Y, p_{X,E})$ and $\mathcal{M}_2 = (\{X, Y\}, \tilde{E}, g, X \leftarrow Y, p_{Y,\tilde{E}})$ with $E \perp\!\!\!\perp X$ and $\tilde{E} \perp\!\!\!\perp Y$ that induce the same joint distribution $p(x, y)$.*

We will prove by construction. Consider the conditional probability transition matrix $\mathbf{Y}|\mathbf{X}$. Without loss of generality, assume $\mathcal{X} = [m], \mathcal{Y} = [n]$. From Lemma 1, it is sufficient to show that there exists $\mathbf{M} \in \mathcal{C}$ and $\mathbf{e}$ such that $vec(\mathbf{Y}|\mathbf{X}) = \mathbf{Me}$.

For now, assume that each entry of $\mathbf{Y}|\mathbf{X}$ is a rational number. Scale the fractional form of each term in $\mathbf{Y}|\mathbf{X}$ so that each denominator becomes the same as the least common multiple of denominators. Denote this least common multiple by λ. Let ϵ be $1/\lambda$.

Let $\mathbf{G}_0 = \mathbf{Y}|\mathbf{X}$ and apply the following procedure for i from 1 to λ in order to construct $\{F_i, i \in [\lambda]\}$: Set $\mathbf{G}_{i+1}$ to $\mathbf{G}_i - \epsilon F_i$, where F_i is the $n \times m, \{0,1\}$ matrix containing only a single 1 in the largest entry of every column of current $\mathbf{G}_i$. This procedure is called to be successful if $\mathbf{G}_i$ is set to all zero matrix for $i = \lambda$.

The above procedure iteratively removes 1 from the numerator of the fractional form of one probability value per column (of $\mathbf{Y}|\mathbf{X}$). Since each column sums to 1, numerators of each column sum to λ. Thus the procedure is successful. Thus we have,

$$\mathbf{Y}|\mathbf{X} = \sum_{k=1}^{\lambda} \epsilon F_k. \tag{A.4}$$

Let $\mathbf{e} = [\epsilon, \epsilon, ..., \epsilon]$ be a length-λ vector. Each entry of $\mathbf{Y}|\mathbf{X}$ is a subset sum of $\mathbf{e}$. Also, since F_i contains a single 1 per column by construction, every subset of $\mathbf{e}$ is disjoint within a column. Thus, we can construct $\mathbf{M} \in \mathcal{C}$ such that $vec(\mathbf{Y}|\mathbf{X}) = \mathbf{M}\mathbf{e}$.

If the entries are not fractional, we can still find small enough ϵ to complete the above procedure.

The same process can be implemented with $\mathbf{X}|\mathbf{Y}$ to obtain $\tilde{E}, g$ such that $X = Y(g, \tilde{E})$.

A.3 Proof of Lemma 2

We prove by construction for any given joint probability distribution. Consider the decomposition described in the proof of Lemma 12. Assume without loss of generality that $\mathcal{X} = \mathcal{Y} = [n]$. Now, instead of taking out $\lambda = 1/\epsilon$, at step i, remove minimum of the maximum probability values at each column of the current $\mathbf{G}_i$ matrix from the maximum probability locations. Thus, at each iteration i, at least one entry of the matrix $\mathbf{G}_i$ is zeroed out. Since sum of the values in each column remains the same after each iteration, after at most $n(n-1)$ steps, each column must have the same single nonzero value. Thus the algorithm finalizes in $n(n-1)+1$ steps.

Notice that this algorithm is the same as the entropy minimization algorithm we propose in Algorithm 1. For a more detailed explanation of the algorithm steps, see Algorithm 1.

In the case when the matrix has zero entries, it can be shown that one can always find a decomposition with $nnz - 1$ terms, where nnz is the number of non-zero elements in the matrix.

A.4 Proposition 5

Proposition 5. *Let the columns of* $\mathbf{Y}|\mathbf{X}$ *be* n *points independently sampled from the uniform distribution over the* $n-1$ *simplex. Then, with probability 1,* $\nexists(\mathbf{M}, \mathbf{e})$ *with* $vec(\mathbf{Y}|\mathbf{X}) = \mathbf{Me}$ *for* $\mathbf{M} \in \{0,1\}^{n^2 \times m}$*, when* $m < n(n-1)$.

Proof. We use the following technique to generate uniformly randomly sampled

points on the simplex in n dimensions [102]:

Lemma 13. *Let $x_i \sim \mathcal{U}[0,1]$ for $i \in [n-1]$ be i.i.d random variables, ordered such that $x_i \geq x_j$ for $i > j$. Let $x_0 = 0$ and $x_n = 1$. Then $\mathbf{u} = [u_i]_{i\in[n]}$ where $u_i = x_i - x_{i-1}, \forall i \in [n]$, is a random vector uniformly distributed over $n-1$ simplex.*

First, we construct $vec(\mathbf{Y}|\mathbf{X})$ directly using $n(n-1)$ uniform i.i.d. random variables: Consider $\tilde{x}_i$ for $i \in [n^2]$ where each consecutive block $\{\tilde{x}_{jn+1}, \tilde{x}_{jn+2}, \ldots, \tilde{x}_{jn+n}\}, j \in \{0, 1, \ldots, n-1\}$ of size n is sampled from the generative model in Lemma 13 before reordering. Thus $\tilde{x}_i$ are i.i.d. with $\tilde{x}_i \sim \mathcal{U}[0,1]$ for $n(n-1)$ indexes by construction. Order $\tilde{x}_i$ within each block in ascending order to get x_i in accordance with Lemma 13. Defining the vectors $\mathbf{x} = [x_i]$ and $\tilde{\mathbf{x}} = [\tilde{x}_i]$, we have $\mathbf{x} = \mathbf{P}\tilde{\mathbf{x}}$ for some permutation matrix $\mathbf{P}$. From $\mathbf{x}$, we can construct $\tilde{\mathbf{z}} = vec(\mathbf{Y}|\mathbf{X})$ using the same map used in Lemma 13 for each block (the map from u_i from x_i in Lemma 13). This construction is a linear map $\mathbf{H}$ where each submatrix of $\mathbf{H}$ is full rank.

For the sake of contradiction, let $\tilde{\mathbf{z}} = \tilde{\mathbf{M}}\mathbf{e}$ be a decomposition where $\tilde{\mathbf{M}} \in \{0,1\}^{n(n-1)\times m}$ where $m < n(n-1)$. Ignoring the entries where $\tilde{z}_i = 1 - x_{i-1}$, we can relabel $\tilde{\mathbf{z}}$ to get $\mathbf{z}$ with $\mathbf{z} \in [0,1]^{n(n-1)}$. Correspondingly, we can write $\mathbf{z} = \mathbf{M}\mathbf{e}$, where $\mathbf{M}$ is the submatrix of $\tilde{\mathbf{M}}$ obtained by ignoring the corresponding rows.

The construction of $\mathbf{z}$ from $\mathbf{x}$ based on the above construction yields $\mathbf{z} = \mathbf{W}\mathbf{x}$ for a full rank matrix $\mathbf{W}$. Notice that any subset of rows are linearly

independent due to specific structure of $\mathbf{H}$.

Let r be the rank of $\mathbf{M}$. Clearly $r < n(n-1)$. Then, some of the (at least $n(n-1)-r$) rows of $\mathbf{M}$ can be written as a unique linear combination of r linearly independent rows.

Consider one such row m_0, where $m_0 = \sum_{i=1}^{r} \alpha_i m_i$ and m_i are a set of linearly independent rows of $\mathbf{M}$. Define $\mathbf{a} = [-1, \alpha_1, \alpha_2, ...\alpha_r]^T$. Take $r+1$ rows of $\mathbf{W}$ corresponding to the selected z_i's to form $\mathbf{W}_r$. Then we have, $\mathbf{a}^T\mathbf{W}_r\mathbf{x} = \mathbf{a}^T\mathbf{W}_r\mathbf{P}\tilde{\mathbf{x}} = 0$. Recall that any subset of rows of $\mathbf{W}$ is full rank, and $\mathbf{P}$ is a permutation matrix. Hence, left nullspace of $\mathbf{W}_r\mathbf{P}$ is empty, implying that $\tilde{\mathbf{x}}$ has to be orthogonal to the vector $\mathbf{a}^T\mathbf{W_r}\mathbf{P} \neq 0$. This is a probability 0 event for any nonzero $\mathbf{a}$, since each $\tilde{x_i}$ is independently sampled from a continuous distribution. Probability that all such constraints are satisfied is zero since it is less than the probability that one particular constraints is satisfied. This argument holds for any fixed $\mathbf{M}$. Since there are finitely many such $\{0,1\}$ matrices, probability that there exists such an $\mathbf{M}$ is 0.

Thus, unless $\mathbf{M}$ has rank at least $n(n-1)$, there does not exist a decomposition $\mathbf{Me} = \mathbf{z}$ with probability 1. □

A.5 Proof of Theorem 1

For sampling from the simplex, we use the following model:

Lemma 14 ([102]). *Let x_i for $i \in [n]$ be independent, exponential random variables with mean 1. Then the random vector $\left[\frac{x_1}{\sum_i x_i}, \frac{x_2}{\sum_i x_i}, ..., \frac{x_n}{\sum_i x_i}\right]$ is uniformly*

distributed over the $n-1$ *simplex.*

Assume above generative model for sampling the distributions of X and E: Let $\{x_i, i \in [n]\}$ and $\{e_i, i \in [\theta]\}$ be sets of independent identically distributed exponential random variables with mean 1. Assign $\mathbb{P}(X = i) = \frac{x_i}{\sum_j x_j}$, and $\mathbb{P}(E = k) = \frac{e_k}{\sum_j e_j}$.

Let $\mathbf{p} = \mathbf{vec}(\mathbf{Y}|\mathbf{X})$. Then, as shown in Section 2.3.1, we can write

$$\mathbf{p} = \mathbf{M}\mathbf{e} \tag{A.5}$$

Notice that jth block of n rows of $\mathbf{p}$ give the conditional probability distribution of Y given $X = j$. Let $S_{i,j}$ represent the set of indices of $\mathbf{e}$ that contribute to the probability of observing $Y = i$ given $X = j$, i.e., $p_{i,j} = \frac{1}{\sum_j e_j} \sum_{k \in S_{i,j}} e_k$. Thus ith row in jth block, or equivalently $i + (j-1)n$th row of $\mathbf{M}$ is 1 in the indices $S_{i,j}$.

The fact that f is generic implies each row of $\mathbf{M}$ is distinct and non-empty.

For the sake of contradiction, assume that there exists $\tilde{E} \perp\!\!\!\perp Y$ with cardinality $m < n(n-1)$, with some deterministic function g such that $X = g(Y, \tilde{E})$ induces the same joint distribution $p(x, y)$. By Lemma 1, this implies that there exists $\tilde{\mathbf{M}} \in \{0,1\}^{n^2 \times m}$ end $\tilde{\mathbf{e}}$ such that

$$\mathbf{q} = \tilde{\mathbf{M}}\tilde{\mathbf{e}}, \tag{A.6}$$

where $\mathbf{q} = \mathbf{vec}(\mathbf{X}|\mathbf{Y})$ and $\tilde{\mathbf{e}}$ is the probability distribution vector of $\tilde{E}$.

Let $q_{i,j} = \mathbb{P}(X = i|Y = j)$. Then from Bayes' rule, we have

$$q_{i,j} = \frac{p_{j,i}x_i}{\sum_i p_{j,i}x_i} = \frac{x_i \sum_{k\in S_{j,i}} e_k}{\sum_i x_i \sum_{k\in S_{j,i}} e_k}. \tag{A.7}$$

Notice that the denominators $\sum_j x_j$ and $\sum_j e_j$ in each term disappear due to cancellation of numerator and denominator.

Thus we have

$$\mathbf{q} = \left[\frac{p_{11}x_1}{\sum_k p_{1k}x_k}, \frac{p_{12}x_2}{\sum_k p_{1k}x_k}, \cdots, \right. \tag{A.8}$$

$$\left. \frac{p_{1n}x_n}{\sum_k p_{1k}x_k}, \frac{p_{21}x_1}{\sum_k p_{2k}x_k}, \cdots \frac{p_{nn}x_n}{\sum_k p_{nk}x_k} \right]^T \tag{A.9}$$

From (A.6), drop the rows of $\mathbf{q}$ that contain x_n in the numerator as well as the corresponding rows of $\tilde{\mathbf{M}}$. The new linear system becomes:

$$\bar{\mathbf{q}} = \bar{\mathbf{M}}\tilde{\mathbf{e}}, \tag{A.10}$$

where $\bar{\mathbf{q}}$ and $\bar{\mathbf{M}}$ are the desribed submatrices of $\mathbf{q}$ and $\tilde{\mathbf{M}}$ respectively.

$\bar{\mathbf{M}}$ has $n(n-1)$ rows and m columns. We have

$$\text{rank}(\bar{\mathbf{M}}) \leq m < n(n-1). \tag{A.11}$$

Since rank of $\bar{\mathbf{M}}$ is less than $n(n-1)$, the rows of $\bar{\mathbf{M}}$ are linearly dependent. This implies there is at least one set of coefficients $\{\alpha_i\}$ not identically zero that satisfies $\alpha\bar{\mathbf{M}} = 0$, where

$$\alpha = \left[\alpha_{1,1}, \quad \alpha_{1,2}, \quad \cdots, \quad \alpha_{1,n-1}, \quad \alpha_{2,1}, \quad \cdots, \quad \alpha_{n,n-1}\right]. \tag{A.12}$$

Then,

$$\alpha\bar{\mathbf{q}} = \alpha\bar{\mathbf{M}}\tilde{\mathbf{e}} = 0 \tag{A.13}$$

Hence, the elements of $\bar{\mathbf{q}}$ should satisfy the linear equation. Then this linear equation in terms of $q_{i,j}$ can be written as:

$$\sum_{i=1}^{n} \frac{\sum_{j=1}^{n-1} \alpha_{i,j} p_{i,j} x_j}{\sum_{j=1}^{n} p_{i,j} x_j} = \frac{\sum_{j=1}^{n-1} \alpha_{1,j} p_{1,j} x_j}{\sum_{j=1}^{n} p_{1,j} x_j} + \frac{\sum_{j=1}^{n-1} \alpha_{2,j} p_{2,j} x_j}{\sum_{j=1}^{n} p_{2,j} x_j}$$
$$+ \frac{\sum_{j=1}^{n-1} \alpha_{3,j} p_{3,j} x_j}{\sum_{j=1}^{n} p_{3,j} x_j} + \ldots + \frac{\sum_{j=1}^{n-1} \alpha_{n,j} p_{n,j} x_j}{\sum_{j=1}^{n} p_{n,j} x_j} = 0 \tag{A.14}$$

Slightly abusing the notation, relabel $p_{i,j}$ as $p_{i,j} = \sum_{k \in S_{i,j}} e_k$, since $\sum_k e_k$ terms cancel in the expression above. We know from Section 2.3.1 that $S_{i,j} \cap S_{k,j} = \emptyset$ for $k \neq i$. Additionally, due to the assumption that f is generic, we have $S_{i,j} \neq S_{i,k}$.

To prove contradiction, in the following we show that for any given non-zero α, this equation is non-zero with probability 1.

To show this, we show that after equating the denominators, each term brings a unique monomial. Hence, the result is a polynomial where each $\alpha_{i,j}$ is accompanied with at least one unique monomial. Thus, the polynomial cannot be identically zero, and the probability of choosing a root of this polynomial is zero.

For now, assume that the denominator is finite. Later, we will show denominator is almost surely finite to complete the argument.

Lemma 15. *If x_i and e_i are i.i.d. exponential random variables with mean 1, and the function f is generic, (A.14) holds with probability 0, i.e.,*

$$\mathbb{P}((A.14) \text{ holds }) = 0.$$

Proof. Multiply each term with the denominators of others to get a single fraction. Then, with a finite denominator, numerator should be zero for equation to hold. Define c_1 to be the coefficient of $x_n^{n-1}x_1$ in the numerator after equating the denominators. Since x_n only appears due to terms from the denominator, we have

$$\begin{aligned} c_1 = &\alpha_{1,1}(p_{1,1}p_{2,n}p_{3,n}\dots p_{n,n}) \\ &+ \alpha_{2,1}(p_{1,n}p_{2,1}p_{3,n}\dots p_{n,n}) \\ &+ \dots + \alpha_{n,1}(p_{1,n}p_{2,n}p_{3,n}\dots p_{n,1}) \end{aligned} \tag{A.15}$$

Since $S_{1,1} \neq S_{1,n}$ due to f being generic, we have,

(a) Either $\exists e_i \in S_{1,1}, e_i \notin S_{1,n}$

(b) Or $\exists e_i \notin S_{1,1}, e_i \in S_{1,n}$

Without loss of generality, assume some $e_{i_1} \in S_{i,1}$, after a potential relabeling. This is possible since $S_{i,1}$ are disjoint for different i and non-zero. (Then, as we will see $e_{i_1}^2$ only appears together with $\alpha_{i,1}$ in (A.15)). Similarly, assume some $e_{i_n} \in S_{i,n}$.

Consider the mulitiplier of coefficient $\alpha_{i,1}$. Assume case (a) holds for $S_{i,1}$, i.e., $\exists e_i \in S_{i,1}, e_i \notin S_{i,n}$. Then, $\alpha_{i,1}$ is accompanied with term e_i^2 since

$e_i \in S_{j,n}$ for some $j \neq i$. Also, it is easy to see that no other term contains e_i^2 since $S_{i,1}$ does not appear again and no $S_{j,1}, j \neq i$ contains e_i.

Assume case (b) holds for the multiplier of $\alpha_{i,1}$, i.e., $\exists e_{i_n} \in S_{i,n}, e_{i_n} \notin S_{i,1}$. Then every term except $\alpha_{i,1}$ is accompanied by either e_{i_n} or ${e_{i_n}}^2$, since $p_{i,n}$ appears in every other term.

Above argument implies that every term is different from the rest. This implies that every distinct $\alpha_{i,j}$ is accompanied by a different monomial in the form $x_n^{n-1} x_j \prod_{k \in T_{i,j}} e_k$, for some $T_{i,j} \subset [\theta]$, where $T_{i,j}$ are distinct for different i. Thus, since at least one $\alpha_{i,j}$ is nonzero the resulting polynomial in the numerator is not identically zero. Then the numerator is a non-zero polynomial of the terms $\{x_1, x_2, \ldots, x_n, e_1, e_2, \ldots, e_n\}$. We know that the roots of a non-zero polynomial defined over a compact domain has Lebesque measure zero [19]. Hence probability of numerator being zero is 0.

Then we have,

$$\begin{aligned}
\mathbb{P}((A.14) = 0) = \mathbb{P}(&\text{ Numerator of (A.14)} = 0 \\
&\text{OR Denominator of (A.14)} = \infty) \\
&\leq \mathbb{P}(\text{ Numerator of (A.14)} = 0) \\
&+ \mathbb{P}(\text{ Denominator of (A.14)} = \infty).
\end{aligned}$$

$\mathbb{P}(\text{ Numerator of (A.14)} = 0)$ is shown to be zero by the argument above. We need to argue that the denominator cannot be infinity.

For this, denote denominator random variable to be ξ. We can write

$$\mathbb{P}(\xi < \infty) = 1 - \mathbb{P}(\limsup_{n\to\infty} \varepsilon_n), \tag{A.16}$$

where ε_n is the event that $\{\xi \geq t_n\}$ for a sequence of t_n such that $\lim_{n\to\infty} t_n = \infty$. Pick $t_n = n^2$.

Since ξ is nonnegative, we can apply Markov inequality to get

$$\mathbb{P}(\varepsilon_n) \leq \frac{\mathbb{E}[\xi]}{t_n}. \tag{A.17}$$

Clearly, $\mathbb{E}[\xi] < \infty$. Since $\sum_n \mathbb{P}(\varepsilon_n) = \mathbb{E}[\xi] \sum_n \frac{1}{t_n} < \infty$, applying Borel-Cantelli lemma, we have

$$\mathbb{P}(\limsup_{n\to\infty} \varepsilon_n) = 0, \tag{A.18}$$

which implies $\mathbb{P}(\xi < \infty) = 1$. Thus, we have

$$\mathbb{P}((A.14) = 0) \leq 0 \Rightarrow \mathbb{P}((A.14) = 0) = 0. \tag{A.19}$$

□

Now we can prove the main theorem:

Assume for the sake of contradiction that there exists a pair $(g, \tilde{E})$ that satisfy $X = g(Y, \tilde{E}), \tilde{E} \perp\!\!\!\perp Y$. By Lemma 1, this is equivalent to the statement that there exists pair $(\tilde{\mathbf{M}}, \mathbf{e})$ that satisfy (A.6). Define events $\varepsilon_1(\tilde{\mathbf{M}}_\mathbf{k}, \tilde{\mathbf{e}}) = \{$ Event that $\mathbf{q} = \tilde{\mathbf{M}}_\mathbf{k}\tilde{\mathbf{e}}$ $\}$ and $\varepsilon_2(\tilde{\mathbf{M}}_\mathbf{k}) = \{$ Event that $\alpha_k \mathbf{q} = 0$ $\}$, $\mathbf{M}_\mathbf{k}$ is a fixed $\{0,1\}^{n^2 \times m}$ matrix and α_k is one set of coefficients imposed by

linear dependence of rows of $\tilde{\mathbf{M}}_{\mathbf{k}}$. Notice that here $\mathbf{q}$ is a random vector, determined by $\{x_i\}, i \in [n]$ and $\{e_i\}, i \in [\theta]$. Clearly, ε_1 implies ε_2, thus $\mathbb{P}(\varepsilon_1) \leq \mathbb{P}(\varepsilon_2)$. Now we can write:

$$\begin{aligned}
&\mathbb{P}(\exists(g, \tilde{E}) \text{ such that } X = g(Y, E)) \\
&= \mathbb{P}(\exists(\tilde{\mathbf{M}}, \tilde{\mathbf{e}}) \text{ such that } \mathbf{q} = \tilde{\mathbf{M}}\tilde{\mathbf{e}}) && \text{(A.20)} \\
&= \mathbb{P}(\exists(\tilde{\mathbf{M}}, \tilde{\mathbf{e}}) \text{ such that } \varepsilon_1 \text{ is true}) && \text{(A.21)} \\
&\leq \mathbb{P}(\exists(\tilde{\mathbf{M}}, \tilde{\mathbf{e}}) \text{ such that } \varepsilon_2 \text{ is true}) && \text{(A.22)} \\
&= \mathbb{P}(\exists(\tilde{\mathbf{M}}) \text{ such that } \varepsilon_2 \text{ is true}) && \text{(A.23)} \\
&\leq \sum_k \mathbb{P}(\text{ Given } \tilde{\mathbf{M}}_{\mathbf{k}}, \alpha_k \mathbf{q} = 0) && \text{(A.24)} \\
&= \sum_k \mathbb{P}(\alpha_k \mathbf{q} = 0) = 0. && \text{(A.25)}
\end{aligned}$$

(A.20) follows from the fact that both representations are equivalent by Lemma 1. (A.22) is due to the fact that if ε_1, then ε_2. (A.23) is due to the fact that ε_2 does not depend on $\tilde{\mathbf{e}}$, but only on $\tilde{\mathbf{M}}$. (A.24) follows from union bound over all matrices $\tilde{\mathbf{M}}_{\mathbf{k}}$. The last equation follows from Lemma 15 and the fact that there are finitely many $\tilde{\mathbf{M}}_{\mathbf{k}} \in \{0,1\}^{n^2 \times m}$ with $m < n(n-1)$ columns. □

A.6 A counterexample when $S_{i,j} = S_{i,k}$

Following counterexample shows that without the additional assumption, identifiability result in Theorem 1 does not hold.

Consider the following equation:

$$\begin{pmatrix} p_{1,1} \\ p_{2,1} \\ p_{3,1} \\ p_{1,2} \\ p_{2,2} \\ p_{3,2} \\ p_{1,3} \\ p_{2,3} \\ p_{3,3} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \times \begin{pmatrix} e_1 \\ e_2 \\ e_3 \end{pmatrix} \tag{A.26}$$

In the reverse direction, we can fit the following system, which has smaller cardinality for the exogenous variable:

$$\begin{pmatrix} q_{1,1} \\ q_{2,1} \\ q_{3,1} \\ q_{1,2} \\ q_{2,2} \\ q_{3,2} \\ q_{1,3} \\ q_{2,3} \\ q_{3,3} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \times \begin{pmatrix} \frac{x_1}{x_1+x_2+x_3} \\ \frac{x_2}{x_1+x_2+x_3} \\ \frac{x_3}{x_1+x_2+x_3} \end{pmatrix} \tag{A.27}$$

Notice that e_i terms completely disappear in this symmetric case. Thus, exogenous variable with n states is sufficient to describe the reverse conditional probability distribution matrix, independent from the cardinality of exogenous variable in the true direction, i.e., the value of θ.

Our main theorem suggests, under the condition that no $S_{i,j}$ is the exact subset of $\{e_1, e_2, ...\}$, no such case can arise, and the reverse direction requires exogenous variable with at least $n(n-1)$ states.

A.7 A counterexample when $p_{i,j} \geq 0$

The critical component of the proof was that each linear equation implied by the rank deficiency of the system had unique non-zero coefficients. Here, we provide a counterexample to the theorem when this condition is violated.

Consider the following system:

$$\begin{pmatrix} p_{1,1} \\ p_{2,1} \\ p_{1,2} \\ p_{2,2} \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{pmatrix} \times \begin{pmatrix} e_1 \\ e_2 \\ e_3 \\ e_4 \end{pmatrix} \tag{A.28}$$

In the reverse direction, we can fit the following system, which has smaller cardinality for the exogenous variable.

$$\begin{pmatrix} q_{1,1} \\ q_{2,1} \\ q_{1,2} \\ q_{2,2} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 1 \\ 1 & 1 \end{pmatrix} \times \begin{pmatrix} \frac{x_1}{x_1+x_2(e_1+e_2)} \\ \frac{x_2(e_1+e_2)}{x_1+x_2(e_1+e_2)} \end{pmatrix} \tag{A.29}$$

Notice that this is true independent of the selection of e_i, hence the theorem cannot be extended to case where $p_{i,j} = 0$ is allowed.

A.8 Proof of Theorem 2

We first define *decomposition problem*, and show it is NP hard.

Definition 11. Decomposition problem: *For a given nonnegative matrix M, with column sums equal to 1, consider the decomposition $\sum_{x \in \mathcal{X}} x F_x$, where F_x are 0,1 matrices with single 1 per column, and $\sum_{x \in \mathcal{X}} x = 1$ with $x \geq 0$. Identify the decomposition that minimizes card($\mathcal{X}$).*

Lemma 16. *The decomposition problem is* NP *hard.*

Proof. We use subset sum problem for the reduction:

Definition 12. Subset sum problem: *For a given set of integers V, and an integer a, decide whether there exists a subset S of V such that $\sum_{u \in S} u = a$.*

Subset sum is a well known NP complete problem. Consider any instance of the subset sum problem with the set $V = \{u_1, u_2, ..., u_m\}$ and an integer a. Assume without loss of generality $u_i \neq 0, \forall i \in V$. If not, one can work with the set of nonzero values in V. Construct the m by 2 matrix $\mathbf{M}$ with $\mathbf{M}(i,1) = u_i$ and $\mathbf{M}(1,2) = a, \mathbf{M}(2,2) = -a + \sum_{i \in [m]} u_i, \mathbf{M}(i,2) = 0, \forall i \in \{2, 3, ..., m\}$.

Update $\mathbf{M}$ by dividing each element by $\sum_{i \in V} u_i$. This does not change the answer to the subset sum problem. Now column sum of $\mathbf{M}$ is 1 for both columns. It can be shown that the decomposition of Lemma 2 can always be applied here to get a decomposition with $|\mathfrak{X}| = m + 1$ (number of non-zero terms -1). We also know that any decomposition need to touch each nonzero element in each column at least once. Hence the column with largest number of nonzero elements yields the lower bound $|\mathfrak{X}^*| \geq m$.

We show that optimal decomposition size is m if and only if there is a subset of V that sums to a.

First, assume $\exists S \subset V$ such that $\sum_{u \in S} u = a$. Consider the following decomposition: Let $x_k = u_k$ for $k \in [m]$. Let $F_k(k,1) = 1$ be the only nonzero element in the first column. Pick the nonzero element in the second column

of F_k based on the membership of u_k in S: Let $F_k(1,2) = 1$ if $u_k \in S$ and $F_k(2,2) = 1$ if $u_k \in V \backslash S$. This decomposition is optimal due to lower bound.

Now we show that if optimum has size m, then $\exists$ a subset sum with value a: Recall that each F_k has a single 1 per column and decomposition has m terms. Since first column of $\mathbf{M}$ has m non-zero terms, each term in the decomposition must be equal to the elements of this column, which is the elements of the given set. Since F_k are 0,1 matrices with single 1 per column by construction, every element of M must be a subset sum of the decomposition terms. Hence, a is a subset sum of set elements.

This shows that a subset sum exists if and only if optimal decomposition has size m. Thus, if we could find the optimal decomposition size in all instances of the decomposition problem, we would solve all instances of the subset sum problem. □

Now, we give a definition related to decomposability:

Definition 13. $\alpha-$decomposability: *A matrix M is called $\alpha-$decomposable if it can be written as a convex combination of α $\{0,1\}$ matrices, each with a single 1 per column.*

Identifying the exogenous variable with minimum cardinality is equivalent to finding minimum size $\mathbf{e}$ with $\mathbf{M} \in \mathcal{C}$ such that $vec(\mathbf{Y}|\mathbf{X}) = \mathbf{Me}$ from Lemma 1. Notice that matrix $\mathbf{Y}|\mathbf{X}$ is $\alpha-$decomposable if and only if there exists $\mathbf{e}$ of size α along with $\mathbf{M} \in \mathcal{C}$ such that $vec(\mathbf{Y}|\mathbf{X}) = \mathbf{Me}$.

Consider any decomposition problem. If we had an algorithm that could solve all instances of the problem of identifying E with minimum support, we could feed the normalized matrix from the decomposition problem as $\mathbf{Y}|\mathbf{X}$ to this algorithm and solve the decomposition problem.

A.8.1 Proof of Theorem 3

Consider m random variables $\{U_1, U_2, \ldots, U_m\}$ each with n states. Let $\{p_1, p_2, \ldots, p_m\}$ stand for the marginal probability distributions of each random variable. Consider the following optimization problem:

$$\begin{aligned} H^*(U_1, U_2, ..., U_m) = \min_{p(u_1,u_2,...,u_m)} \quad & H(U_1, U_2, ..., U_m) \\ \text{s. t.} \quad & p(u_i) = p_i, \forall i \end{aligned} \tag{A.30}$$

In words, optimization problem finds the joint distribution of m variables with minimum entropy, subject to marginal distribution constraints for every variable. Notice that this problem is non-convex: It minimizes a concave function with respect to linear constraints.

Let X, E be discrete, independent random variables where $X \in [m]$ and $E \in [t]$. Recall that every conditional distribution $\mathbb{P}(Y|X = x)$ can be written as the distribution of a function of random variable E. Let us investigate the underlying probability space, in order to generate the probability space to optimize the joint distribution over.

We can consider the product probability space

$$\mathcal{P} = (\Omega = \Omega_X \times \Omega_E = [m] \times [t] \ , \ \mathcal{F} = 2^{\Omega} \ , \ p) \tag{A.31}$$

for random variables $X : \Omega_X \to [n], E : \Omega_E \to [t]$ with $X(\omega) = \omega, Y(\omega) = \omega$. Then for $\omega = (\omega_x, \omega_e) \in \Omega$, $p(\omega) = p_X(\omega_x)p_E(\omega_e)$.

Consider the equation $Y = f(X, E)$. Let $f_x : \Omega_E \to \Omega_Y$ be the function mapping E to Y when $X = x$, i.e., $f_x(E) \coloneqq f(x, E)$. Then

$$\mathbb{P}(Y = y|X = x) = \mathbb{P}(f_x(E) = y|X = x) \tag{A.32}$$

$$= \mathbb{P}(f_x(E) = y), \tag{A.33}$$

where last equality follows from the fact that $X \perp\!\!\!\perp E$.

Let sigma algebra generated by the random variable $f(x, E)$ be $\mathcal{F}_{\omega_x}$. Formally, the sigma algebras generated by $f(x, E)$ are subsigma algebras of disjoint, but identical sigma algebras. In other words, even though $\mathcal{F}_{\omega_x} \subseteq 2^{(\omega_x, \Omega_E)}$ are disjoint for different ω_x, (ω_x, Ω_E) are identical for every $\omega_x \in \Omega_X$. Thus the sigma algebras generated by $f(x, E) = g_x(E)$ can be thought of as subsigma algebras of the same sigma algebra, i.e., the one generated by E. In other words, we can equivalently construct $\mathcal{F}_{\omega_x} \subseteq 2^{\Omega_E}$. This construction allows us to talk about the joint probability distribution of the random variables $\{f_x(E) : x \in \mathcal{X}\}$.

Let $U_i = f_i(E)$. Then we have,

$$H(E) = H(E|U_1, U_2, ..., U_m) + H(U_1, U_2, ..., U_m) \tag{A.34}$$

$$- H(U_1, U_2, ..., U_m|E) \tag{A.35}$$

$$\geq H(U_1, U_2, ..., U_m). \tag{A.36}$$

Inequality follows from the fact that

$$H(U_1, U_2, ..., U_m|E) = 0, \qquad H(E|U_1, U_2, ..., U_m) \geq 0.$$

$H(E) \geq H(U_1, U_2, ..., U_m)$. Thus, the best lower bound on the entropy of exogenous variable E can be obtained by solving the optimization problem below.

$$\begin{aligned} H(E) \geq \quad & \min_{p(u_1,u_2,...,u_m)} \quad H(U_1, U_2, ..., U_m) \\ & \text{subject to} \quad p(u_i) = p_i,\ i = 1, \ldots, m. \end{aligned} \tag{A.37}$$

Since we are picking E with minimum possible entropy without restricting (not observing) the functions f_x, we can actually construct an E that achieves this minimum: Let the optimal joint distribution be $p^*(u_1, u_2, ..., u_m)$. We can construct E with n^m states with state probabilities equal to $p^*(u)$ for each configuration of u. This E has the same entropy as the joint entropy, since probability values are the same. Thus,

$$\begin{aligned} H(E^*) = \quad & \min_{p(u_1,u_2,...,u_m)} \quad H(U_1, U_2, ..., U_m) \\ & \text{subject to} \quad p(u_i) = p_i \end{aligned}$$

The functions f_i, where U_i has the same distribution as $f_i(E)$, can be constructed from the distribution of E and U_i. This determines the function f, hence the causal model $\mathcal{M}$ that induces the conditional distribution $\mathbf{Y}|\mathbf{X}$. Note that $E \perp\!\!\!\perp X$ by construction. (For a complete argument on how one can always generate $E \perp\!\!\!\perp X$ based on a set of samples of X, Y, see the proof of Lemma 1).

A.8.2 Proof of Corollary 2

Assume there exists a black box that could find the causal model with minimum entropy exogenous variable E. Consider an arbitrary instance of the problem of minimizing the joint entropy of a set of variables $\{U_1, U_2, \ldots, U_n\}$ subject to marginal constraints. Construct matrix $\mathbf{M} = [p_1, p_2, \ldots, p_n]$, where p_i is the distribution of variable U_i as a column vector. Feed $\mathbf{M}$ into this black box as a hypothetical conditional distribution $\mathbf{Y}|\mathbf{X}$. From the proof of Theorem 3, $H(E)$ output by this black box gives the minimum entropy joint distribution of the variables U_i. Hence, finding the causal model with minimum entropy would give the solution to this NP hard problem.

A.8.3 Proof of Proposition 2

Given a joint distribution $p(x, y)$, consider the following algorithm: In stage 1, feed the set of conditional distributions $\{\mathbb{P}(Y|X = i) : i \in [n]\}$ to algorithm $\mathcal{A}$ to obtain E with minimum entropy. Algorithm outputs a variable E along with $\{f_1, f_2, \ldots, f_n\}$ such that the distribution of $f_j(E)$ is the same as the conditional distribution of Y given $X = j, \forall j$. This set of f_j determine f where $Y = f(X, E), E \perp\!\!\!\perp X$. Since algorithm optimizes entropy of E, $H(E) \leq H(E_0)$. In stage 2, feed the conditional distributions $\{\mathbb{P}(X|Y = i) : i \in [n]\}$ to algorithm $\mathcal{A}$ to obtain $\tilde{E}$. From the conjecture any $\tilde{E}$ satisfies $H(X) + H(E_0) < H(Y) + H(\tilde{E})$. Since $H(E) \leq H(E_0)$, we have $H(X) + H(E) < H(Y) + H(\tilde{E})$, which can be used for identifiability.

A.8.4 Greedy Entropy Minimization Outputs a Local Optimum

Proposition 6. *For two variables, Algorithm 1 always returns a local optimum.*

Consider random variables U, V with marginal distributions p_u, p_v. We first show that the KKT conditions on the problem (A.30) imply that the optimal solution is *quasi-orthogonal*:

$$p^*(u, v) = \mathbf{M} = \mathbf{U}(x, y)\delta_{x,y}, \tag{A.38}$$

where $\mathbf{U}$ is rank 1 and $\delta_{u,v}$ is an indicator for the support of optimal joint distribution. We call such $\mathbf{M}$ as masked submatrix of a rank 1 matrix.

Let (i, j)th entry of the joint distribution be $x_{i,j}$. We have n^2 variables $\{x_{i,j}, i \in [n], j \in [n]\}$ to optimize over. In a general optimization problem

$$\begin{aligned} \min_{x} \quad & f_0(x) \\ \text{s. t.} \quad & h_i(x) = 0, i \in [p] \\ & f_i(x) \leq 0, i \in [m], \end{aligned}$$

Lagrangian becomes

$$L(x, \lambda, v) = f_0(x) + \sum_{i=1}^{m} \lambda_i f_i(x) + \sum_{i=1}^{p} v_i h_i(x), \tag{A.39}$$

which gives the KKT conditions

$$\begin{aligned} & f_i(x^*) \leq 0, i \in [m] \\ & h_i(x^*) = 0, i \in [p] \\ & \lambda_i^* \geq 0, i \in [m] \\ & \lambda_i^* f_i(x^*) = 0, i \in [m] \\ & \nabla L(x^*, \lambda^*, v^*) = 0 \end{aligned}$$

This implies, for fixed i, either $f_i(x^*) = 0$ or $\lambda_i^* = 0$. The optimization problem we have is

$$\begin{aligned} \min_{x_{i,j}} \quad & \sum_{i,j} -x_{i,j} \log x_{i,j} \\ \text{s. t.} \quad & \sum_j x_{i,j} = p_u(i), \forall i, \quad \sum_i x_{i,j} = p_v(j), \forall j \\ & x_{i,j} \geq 0, \forall i, j. \end{aligned}$$

Substituting corresponding f_i, h_i, we get the following conclusion: At optimal point $x_{i,j}$, either $x_{i,j} = 0$, or $1 - \log x_{i,j} + v_i^{(1)} + v_j^{(2)} = 0$. This implies that $x_{i,j} = 2^{1+v_i^{(1)}+v_j^{(2)}}$. Hence the optimal joint satisfies $p_{i,j}^* = u_i v_j$ for some vectors u, v, whenever $p_{i,j}^* \neq 0$.

Next we show that such u, v can be constructed from any algorithm output:

Theorem 18. *For any matrix* $\mathbf{M}$ *output by Algorithm 1, there exists* u, v *where* $\mathbf{M}$ *is a masked submatrix of* uv^T.

Proof of Proposition 6. Consider the following variant of Algorithm 1 for two distributions $\mathbf{p}, \mathbf{q}$: Initialize an $n \times n$ zero matrix $\mathbf{M}_0$. In every iteration, find $m_1 = \max_i \mathbf{p}(i), m_2 = \max_i \mathbf{q}(i)$. Let $a = \arg\max_i \mathbf{p}(i)$ and $b = \arg\max_i \mathbf{q}(i)$. Assign $r = \min\{m_1, m_2\}$ to the (a, b)th entry of $\mathbf{M}$. Update $\mathbf{p}(a) \leftarrow \mathbf{p}(a) - r, \mathbf{q}(b) \leftarrow \mathbf{q}(b) - r$. Repeat the process until $\mathbf{p} = 0, \mathbf{q} = 0$.

This variant constructs the joint distribution matrix rather than the distribution of E directly. We need the following definitions:

Definition 14. *An ordered set of coordinates* $((i_1, j_1), (i_2, j_2), \ldots, (i_m, j_m))$ *of a matrix* $\mathbf{M}$ *is called a path on matrix* $\mathbf{M}$*, if either* $i_{t+1} = i_t$ *or* $j_{t+1} = j_t, \forall t \in [m-1]$.

Definition 15. *An ordered set of coordinates* $((i_1, j_1), (i_2, j_2), \ldots, (i_m, j_m))$ *of a matrix* $\mathbf{M}$ *is called a cycle on matrix* $\mathbf{M}$*, if either* $i_{t+1} = i_t$ *or* $j_{t+1} = j_t, \forall t \in [m-1]$ *and either* $i_1 = i_m$ *or* $j_1 = j_m$.

First we prove a structural characterization for matrix $\mathbf{M}$.

Lemma 17. *Any matrix* $\mathbf{M}$ *output by Algorithm* 1 *contains no cycles.*

Proof. Consider a bipartite graph G with n left and n right vertices constructed as follows: G has an edge (i, j) if and only if $\mathbf{M}(i, j)$ is nonzero.

It is easy to see that the matrix $\mathbf{M}$ has no cycles if and only if the corresponding bipartite graph G has no cycles. We claim that, for an $\mathbf{M}$ output by Algorithm 1, corresponding G cannot have any cycles.

Construct the bipartite graph in the same order as matrix $\mathbf{M}$ is created by Algorithm 1. Notice that when the algorithm assigns a value to $\mathbf{M}(i, j)$, it satisfies either ith row constraint or jth column constraint. Then no other value can be assigned to that row or column. We call this zeroing out the corresponding row/column.

In the bipartite representation, say an edge (i, j) is added at time t. We call a vertex of the added edge *closed*, if the corresponding dimension (row or column) is zeroed out. Thus each added edge to the bipartite graph closes

one of the endpoints of that edge. A closed vertex cannot participate to the formation of any other edges. This captures the fact that when zeroed out, a row/column cannot be assigned a non-zero value again by the algorithm.

Assume at time t, a cycle is formed in the bipartite graph. Then at time $t-1$, there must be a path with a single edge missing. Let the edges of this path be labeled as $\{e_1, e_2, \ldots e_m\}$. There is a time order in which these edges are constructed. Let e_k be the first edge formed in this path. Then one of its endpoints must be closed. However it belongs to a path, which means an edge was attached to a closed vertex, which is a contradiction. Assume e_k is the edge at the end of the path and the closed endpoint is also the endpoint of the path. However an edge is attached to this closed vertex at time t to form the cycle, which is a contradiction. □

Consider a matrix $\mathbf{M}$ output by the algorithm. We prove that $\mathbf{M}$ is a masked submatrix of uv^T by construction:

Let the first entry selected by Algorithm 1 have coordinates (i_0, j_0). Since algorithm zeroes out either the row or column, (i_0, j_0) is the only non-zero entry in either its column or row. Assume without loss of generality that selection of (i_0, j_0) by Algorithm 1 zeroes out row i_0.

Initialize by assigning $u_{i_0} = 1$ and $v_{j_0} = \mathbf{M}(i_0, j_0)$.

Let S_{j_0} represent the non-zero row indices for column j_0. Now assign the values of $u(k)$ for $k \in S_{j_0}$ such that $u(S_{j_0})v_{j_0} = \mathbf{M}(S_{j_0}, j_0)$, where $u(S)$ stands for the subvector containing entries with index in S.

Repeat the above procedure by exhausting either a row or column at each time.

Lemma 18. *The above construction never runs into an entry (i, j) for which both u_i and v_j were assigned before.*

Proof. Assume otherwise. Then, due to the process of assigning the entries of u, v, there must be two paths from (i, j) to the starting point of (i_0, j_0): One path that follows column j, and one path that follows row i. In other words, there exists $k, l \in [n]$ such that there are two paths $((i, j), (k, j), \ldots, (i_0, j_0))$ and $((i, j), (i, l), \ldots, (i_0, j_0))$. Combining these paths, we get the cycle $((i, j), (k, j), \ldots, (i_0, j_0), \ldots, (i, l))$, which is a contradicton. □

Hence, algorithm selects every element in u and v at most once. Thus it produces a valid assignment for u, v. □

A.8.5 Implementation Details and Sampling Distributions with Diverse Entropy Values

We implemented our algorithms in MATLAB. In order to sample from a wide range of entropy values, when we need to sample a distribution from the $n - 1$ dimensional simplex, we generate n independent identically distributed log Gaussian random variables with parameter σ. For verifying the conjecture on artificial data, we generate the distributions for E swiping the σ parameter from 2 to 8, taking only the integer values. The distribution for X is uniformly randomly sampled over the simplex.

For the true causal direction, the function f is sampled as follows: We generate θ matrices F_i, where each has a single 1 per column, randomly selected out of all n rows. Each F_e represent the function $f_e(X) \coloneqq f(X, e)$. Together with $\mathbf{e}$, this determines the conditional probability distribution matrix $\mathbf{Y}|\mathbf{X}$.

The number of states for quantization is chosen as $n = \min\{N/10, 512\}$, where N is the number of samples for that particular cause-effect pair. Hence, we pick n to assure each state has at least 10 samples on average. An upper bound of 512 is used to limit the computational complexity.

Appendix B

Appendix for Entropic Latent Variable Discovery

B.1 $I(X;Y|Z)$ vs. $H(Z)$radeoff Curve

Figure B.1 shows the $I(X;Y|Z)$-$H(Z)$ tradeoff LatentSearch (Algorithm 2) obtains for a joint distribution sampled as follows: The distribution of Z as well as the conditional distributions $p(X|z), p(Y|z), \forall z$ are chosen uniformly at random over the simplex.

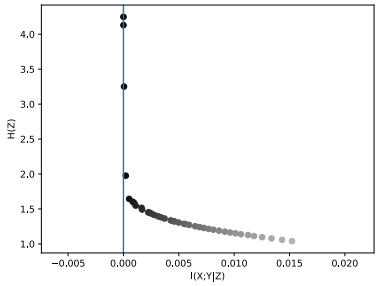


Figure B.1: $I(X;Y|Z)$ vs. $H(Z)$ tradeoff curve obtained by LatentSearch (Algorithm 2) for an arbitrary joint $p(x,y)$ from the graph $X \leftarrow Z \rightarrow Y$. We observed that the curve's shape is consistent across many runs irrespective of the graph, although the crossing point where $I(X;Y|Z) = 0$ changes.

B.2 Proof of Theorem 4

We write the objective function more explicitly in terms of the optimization variables $q(z|x,y)$:

$$\mathcal{L}(q(\cdot|\cdot,\cdot)) = \sum_{x,y,z} q(x,y,z)\log\left(\frac{q(x,y|z)}{q(x|z)q(y|z)}\right) - \beta\sum_{z} q(z)\log(q(z)) \tag{B.1}$$

$$= \sum_{x,y,z} p(x,y)q(z|x,y)\log\left(\frac{q(z|x,y)}{q(z|x)q(z|y)}\right) + I(X;Y) \tag{B.2}$$

$$+ (1-\beta)\sum_{z} q(z)\log(q(z)), \tag{B.3}$$

by Bayes rule and assuming that $q(z|x,y)$ and $p(x,y)$ are strictly positive.

Our objective then is

$$\begin{aligned} \underset{q(z|x,y)}{\text{minimize}} \quad & \mathcal{L}(q(z|x,y)) \\ \text{subject to} \quad & \sum_{z} q(z|x,y) = 1,\ \forall x,y, \\ & q(z|x,y) \geq 0,\ \forall z,x,y. \end{aligned} \tag{B.4}$$

We can write the Lagrangian, which we represent with $\bar{\mathcal{L}}$, as

$$\bar{\mathcal{L}} = \sum_{x,y,z} p(x,y)q(z|x,y)\log\left(\frac{q(z|x,y)}{q(z|x)q(z|y)}\right) + I(X;Y) \tag{B.5}$$

$$+(1-\beta)\sum_{z} q(z)\log(q(z)) + \sum_{x,y}\delta_{x,y}\left(\sum_{z} q(z|x,y) - 1\right) \tag{B.6}$$

In order to find the stationary points of the loss, we take its first derivative and set it to zero. To compute the partial derivatives, notice that $q(z|x), q(z|y), q(z)$ are linear functions of $q(z|x,y)$ (use Bayes rule and

marginalization). We can then easily write the partial derivatives of these quantities with respect to $q(z|x,y)$ as follows:

$$\begin{aligned}
\frac{\partial q(z|x)}{\partial q(z|x,y)} &= \frac{\partial \sum_{y'} q(z|x,y')p(y'|x)}{\partial q(z|x,y)} = p(y|x), \\
\frac{\partial q(z|y)}{\partial q(z|x,y)} &= \frac{\partial \sum_{x'} q(z|x',y)p(x'|y)}{\partial q(z|x,y)} = p(x|y) \\
\frac{\partial q(z)}{\partial q(z|x,y)} &= \frac{\partial \sum_{x',y'} q(z|x',y')p(x',y')}{\partial q(z|x,y)} = p(x,y).
\end{aligned}$$

Using these expressions we have the following.

$$\begin{aligned}
\frac{\partial \bar{\mathcal{L}}}{\partial q(z|x,y)} &= p(x,y)\left[1 + \log(q(z|x,y)) - (1 + \log(q(z|x))) - (1 + \log(q(z|y)))\right. \\
&\qquad \left. + (1-\beta)(1 + \log(q(z))) + \delta_{x,y}\right] \\
&= p(x,y)\left[-\beta + \delta_{x,y} + \log\left(\frac{q(z|x,y)q(z)^{1-\beta}}{q(z|x)q(z|y)}\right)\right]
\end{aligned}$$

Assuming $p(x,y) > 0$, any stationary point then satisfies

$$q(z|x,y) = \left(\frac{1}{2}\right)^{\beta - \delta_{x,y}} \frac{q(z)^{1-\beta}}{q(z|x)q(z|y)}. \tag{B.7}$$

Since $q(z|x,y)$ is a probability distribution, we have

$$\sum_z q(z|x,y) = \left(\frac{1}{2}\right)^{\beta - \delta_{x,y}} \sum_z \frac{q(z)^{1-\beta}}{q(z|x)q(z|y)} = 1 \tag{B.8}$$

Defining $N(x,y) \coloneqq \left(\frac{1}{2}\right)^{\beta - \delta_{x,y}}$, we have

$$N(x,y) = \frac{1}{\sum_z \frac{q(z)^{1-\beta}}{q(z|x)q(z|y)}}. \tag{B.9}$$

From the algorithm description, any stationary point of Algorithm 2 should satisfy

$$q(z|x,y) = \frac{1}{N(x,y)} \frac{q(z|x)q(z|y)}{q(z)^{1-\beta}}, \tag{B.10}$$

for the same $N(x,y)$ defined above. Therefore a point is a stationary point of the loss function if and only if it is a stationary point of LatentSearch (Algorithm 2). □

B.3 Proof of Theorem 5

We can rewrite the loss as

$$\mathcal{L}(q(\cdot|\cdot,\cdot)) = \sum_{x,y,z} q(x,y,z) \log\left(\frac{q(x,y|z)}{q(x|z)q(y|z)}\right) - \beta \sum_{z} q(z)\log(q(z)) \tag{B.11}$$

$$= \sum_{x,y,z} p(x,y)q(z|x,y) \log\left(\frac{q(z|x,y)}{q(z|x)q(z|y)}\right) + I(X;Y) \tag{B.12}$$

$$+ (1-\beta)\sum_{z} q(z)\log(q(z)), \tag{B.13}$$

If we substitute $\beta = 1$, we obtain

$$\mathcal{L}(q(\cdot|\cdot,\cdot)) = \sum_{x,y,z} p(x,y)q(z|x,y) \log\left(\frac{q(z|x,y)}{q(z|x)q(z|y)}\right) + I(X;Y). \tag{B.14}$$

Our objective is

$$\begin{aligned} \underset{q(z|x,y)}{\text{minimize}} \quad & \mathcal{L}(q(z|x,y)) \\ \text{subject to} \quad & \sum_{z} q(z|x,y) = 1, \forall x,y. \end{aligned} \tag{B.15}$$

Notice that $\mathcal{L}(q(z|x,y))$ is not convex or concave in $q(z|x,y)$. However we can

rewrite the minimization as follows:

$$\begin{aligned}\underset{q(z|x,y)}{\text{minimize}} \quad & \underset{r(z|x),s(z|y)}{\text{minimize}} \quad \sum_{x,y,z} p(x,y)q(z|x,y)\log\left(\frac{q(z|x,y)}{r(z|x)s(z|y)}\right) + I(X;Y) \\ & \text{subject to} \quad \sum_{z} q(z|x,y) = 1, \forall x, y \\ & \qquad\qquad \sum_{z} r(z|x) = 1, \forall x, \\ & \qquad\qquad \sum_{z} s(z|y) = 1, \forall y.\end{aligned} \tag{B.16}$$

To see that (B.16) is equivalent to (B.15), notice that the optimum for the inner minimization is $r^*(z|x) = q(z|x)$ and $s^*(z|x) = q(z|y)$. This is due to the fact that (B.16) is convex in $r(z|x)$ and $s(z|y)$ and concave in $t(z)$, which can be seen through the partial derivatives of the Lagrangian:

$$\underset{q(z|x,y)}{\text{minimize}} \quad \underset{r(z|x),s(z|y)}{\text{minimize}} \quad \sum_{x,y,z} p(x,y)q(z|x,y)\log\left(\frac{q(z|x,y)}{r(z|x)s(z|y)}\right) + I(X;Y) \tag{B.17}$$

$$+ \sum_{x,y} \delta_{x,y}\left(\sum_{z} q(z|x,y) - 1\right) + \sum_{x} \eta_x \left(\sum_{z} r(z|x) - 1\right) \tag{B.18}$$

$$\underbrace{+ \sum_{x} \nu_y \left(\sum_{z} s(z|y) - 1\right)}_{\bar{\mathcal{L}}} \tag{B.19}$$

For fixed $q(z|x,y), s(z|y)$, we have

$$\begin{aligned}\frac{\partial \bar{\mathcal{L}}}{\partial r(z|x)} &= -\frac{p(x)q(z|x)}{r(z|x)} + \eta_x \\ \frac{\partial^2 \bar{\mathcal{L}}}{\partial r(z|x)^2} &= \frac{p(x)q(z|x)}{r(z|x)^2}.\end{aligned}$$

Therefore $\bar{\mathcal{L}}$ is convex in $r(z|x)$ and the optimum can be obtained by setting the first derivative to zero. Then we have

$$r^*(z|x) = \frac{p(x)q(z|x)}{\eta_x}, \forall x, z. \tag{B.20}$$

Since we have $\sum_z r^*(z|x) = \frac{p(x)}{\eta_x} \sum_z q(z|x) = 1$, we obtain $r^*(z|x) = q(z|x)$. Similarly, we can show that $s^*(z|x) = q(z|y)$. Notice that this inner minimization is exactly the same as the first update of Algorithm 2.

We can also show that $\mathcal{L}$ is convex in the variables r, s jointly: This can be seen through the fact that $\frac{\partial^2}{\partial r(z|x)s(z|y)}\mathcal{L} = 0$ and the Hessian is positive definite.

This concludes that (B.16) is equivalent to (B.4). Moreover, since the objective function is convex in $q(z|x,y)$ and also jointly convex in $r(z|x), s(z|y)$, we can switch the order of the minimization terms. Therefore, we can equivalently write

$$\underset{r(z|x),s(z|y)}{\text{minimize}} \quad \underset{q(z|x,y)}{\text{minimize}} \quad \sum_{x,y,z} p(x,y)q(z|x,y) \log\left(\frac{q(z|x,y)}{r(z|x)s(z|y)}\right) + I(X;Y) \tag{B.21}$$

$$+ \sum_{x,y} \delta_{x,y} \left(\sum_z q(z|x,y) - 1\right) + \sum_x \eta_x \left(\sum_z r(z|x) - 1\right) \tag{B.22}$$

$$\underbrace{+ \sum_x \nu_y \left(\sum_z s(z|y) - 1\right)}_{\bar{\mathcal{L}}}. \tag{B.23}$$

Let us analyze the inner minimization in this equivalent formulation for fixed $r(z|x), s(z|x)$. Similarly, we can take the partial derivative as follows:

$$\begin{aligned}\frac{\partial \bar{\mathcal{L}}}{\partial q(z|x,y)} &= p(x,y)\left[1+\log(q(z|x,y))-\log(r(z|x))-\log(s(z|x))+\delta_{x,y}\right] \\ &= p(x,y)\left[1+\delta_{x,y}+\log\left(\frac{q(z|x,y)}{r(z|x)s(z|y)}\right)\right] \\ \frac{\partial^2 \bar{\mathcal{L}}}{\partial q(z|x,y)^2} &= p(x,y)\left[\frac{1}{q(z|x,y)}\right].\end{aligned}$$

Notice that $\frac{\partial^2 \bar{\mathcal{L}}}{\partial q(z|x,y)^2} > 0$. Hence $\bar{\mathcal{L}}$ is convex in $q(z|x,y)$. Then the optimum can be obtained by setting the first derivative to zero. We have

$$p(x,y)\left[1+\delta_{x,y}+\log\left(\frac{q(z|x,y)}{r(z|x)s(z|y)}\right)\right]=0, \tag{B.24}$$

or equivalently

$$q(z|x,y)=\left(\frac{1}{2}\right)^{1+\delta_{x,y}} r(z|x)s(z|y). \tag{B.25}$$

Note that if we define

$$N(x,y) \coloneqq \sum_z r(z|x)s(z|y),$$

since $\sum_z q(z|x,y) = \left(\frac{1}{2}\right)^{1+\delta_{x,y}} \sum_z r(z|x)s(z|y) = 1$, we can write

$$q(z|x,y)=\frac{1}{N(x,y)} r(z|x)s(z|y). \tag{B.26}$$

This is exactly the same as the second update of LatentSearch (Algorithm 2) if $r(z|x) = q(z|x), s(z|y) = q(z|y)$.

Therefore, if $q_i(z|x,y)$ is the current conditional at iteration i, the next update of LatentSearch (Algorithm 2) is equivalent to first solving the inner

minimization of (B.16) thereby assigning $r(z|x) = q_i(z|x), s(z|y) = q_i(z|y)$, then switching the order of the minimization operations, and solving the inner minimization of (B.21), therefore assigning $q_{i+1}(z|x,y) = \frac{1}{N(x,y)} q_i(z|x) q_i(z|y)$. In each of this two-step optimization iteration, either loss function goes down, or it does not change. If it does not change, the algorithm has converged. Otherwise, it cannot go down indefinitely since loss (3.1) is lower bounded as $I(X;Y|Z) \geq 0$ and $H(Z) \geq 0$ and therefore has to converge. This proves convergence of the algorithm to either a local minimum or a saddle point. The converged point cannot be a local maximum since it is arrived at after a minimization step. □

B.4 Proof of Theorem 6

Since X, Y are discrete variables, we can represent the joint distribution of X, Y in matrix form. Let $\mathbf{M} = [p(x,y)]_{(x,y)\in[m]\times[n]}$. With a slight abuse of notation, let $\mathbf{z} \coloneqq [z_1, z_2, \dots z_k]$ be the probability mass (row) vector of variable Z, i.e., $\mathbb{P}[Z = i] = \mathbf{z}[i] = z_i$. Similarly, let $\mathbf{x}_z \coloneqq [x_{z,1}, x_{z,2}, \dots x_{z,k}]$ be the conditional probability mass vector of X conditioned on $Z = z$, i.e., $\mathbb{P}[X = i|Z = z] = \mathbf{x}_z[i] = x_{z,i}$. Finally, let $\mathbf{y}_{z,x} \coloneqq [y_{z,x,1}, y_{z,x,2}, \dots y_{z,x,n}]$ be the conditional probability mass vector of Y conditioned on $X = x$ and $Z = z$. We can write the matrix $\mathbf{M}$ as follows:

$$\mathbf{M} = \sum_{i=1}^{k} z_i \begin{bmatrix} x_{i,1}\mathbf{y}_{i,1} \\ x_{i,2}\mathbf{y}_{i,2} \\ \vdots \\ x_{i,m}\mathbf{y}_{i,m} \end{bmatrix} \tag{B.27}$$

Now suppose for the sake of contradiction that there exists such a $q(x, y, z)$ such that $\sum_z q(x, y, z) = p(x, y)$ and $\perp\!\!\!\perp XYZ$. Then $\mathbf{M}$ admits a factorization of the form

$$\mathbf{M} = \sum_{i=1}^{k} z'_i \begin{bmatrix} x'_{i,1}\mathbf{y}'_{i,1} \\ x'_{i,2}\mathbf{y}'_{i,2} \\ \vdots \\ x'_{i,m}\mathbf{y}'_{i,m} \end{bmatrix}, \tag{B.28}$$

where $x'_{i,j}, \mathbf{y}'_{i,j}, z'_i$ are due to the joint $q(x, y, z)$ and are potentially different form their counterparts in (B.27). Notice that since $\perp\!\!\!\perp XYZ$, we have $\mathbf{y}'_{i,j} = \mathbf{y}'_{i,l}, \forall (j, l) \in [k] \times [m]$. Therefore the matrices

$$\begin{bmatrix} x'_{i,1}\mathbf{y}'_{i,1} \\ x'_{i,2}\mathbf{y}'_{i,2} \\ \vdots \\ x'_{i,m}\mathbf{y}'_{i,m} \end{bmatrix}, \tag{B.29}$$

are rank 1 $\forall i \in [k]$. Therefore, $\mathbf{M}$ has NMF rank at most k. Since matrix rank is upper bounded by the NMF rank, $\text{rank}(M) \leq k$. Therefore, there exists a $q(x, y, z)$ such that $\sum_z q(x, y, z) = p(x, y)$ and $\perp\!\!\!\perp XYZ$ *only if* $\text{rank}(M) \leq k$. Next, we show that under the generative model described in the theorem statement, this happens with probability zero.

We have the following lemma:

Lemma 19. *Let $\{\mathbf{x}_i : i \in [n]\}$ be a set of vectors sampled independently, uniformly randomly from the simplex S_{n-1} in n dimensions. Then, $\{\mathbf{x}_i : i \in [n]\}$ are linearly independent with probability 1.*

Proof. If $\mathbf{x}_i$ are linearly dependent, then there exists a set $\{\alpha_i : i \in [n]\}$ such that $\sum_{i=1}^{n} \alpha_i \mathbf{x}_i = 0$. Let $j = \arg\max\{i \in [n] : \alpha_i > 0\}$. Equivalently $\mathbf{x}_j$ is in the range of the set of vectors $\{\mathbf{x}_i : i \in [j-1]\}$. Therefore, we can write

$$\mathbb{P}\left[\{\mathbf{x}_i : i \in [n]\} \text{ are linearly independent }\right] \leq \sum_{i=2}^{n} \mathbb{P}\left[\mathbf{x}_i \in R(\mathbf{x}_1, \ldots, \mathbf{x}_{i-1})\right], \tag{B.30}$$

where $R(\mathbf{x}_1, \ldots, \mathbf{x}_{i-1})$, is the range of the vectors $\mathbf{x}_1, \ldots, \mathbf{x}_{i-1}$, i.e., the vector space spanned by $\mathbf{x}_1, \ldots, \mathbf{x}_{i-1}$.

Notice that $\dim(R(\mathbf{x}_1, \ldots, \mathbf{x}_{i-1})) < n-1, \forall i \leq n-1$. Therefore, codimension of $R(\mathbf{x}_1, \ldots, \mathbf{x}_{i-1})$ with respect to the simplex is non-zero $\forall i \leq n-1$. Therefore, the Lebesgue measure of $R(\mathbf{x}_1, \ldots, \mathbf{x}_{i-1}) \cap S_{n-1}$ is zero with respect to the uniform measure over S_{n-1}. Hence, $\mathbb{P}\left[\mathbf{x}_i \in R(\mathbf{x}_1, \ldots, \mathbf{x}_{i-1})\right] = 0, \forall i \leq n-1$.

The above argument does not hold for the last term in the summation in (B.30). However, intersection of any $n-1$ dimensional vector space with the simplex S_{n-1} is an $n-2$ dimensional slice of the simplex [137]. Therefore, it has Lebesgue measure zero with respect to the uniform measure over the simplex. □

Corollary 6. *Let $\{\mathbf{x}_i : i \in [n]\}$ be a set of vectors sampled independently, uniformly randomly from the simplex S_{n-1} in n dimensions. Let $\{c_i \neq 0 : i \in [n]\}$ be arbitrary real scalars that are non-zero. Then, $\{c_i \mathbf{x}_i : i \in [n]\}$ are linearly independent with probability 1.*

Proof. The proof of Lemma 19 goes through since the span of a set of vectors does not change with scaling of the vectors. □

$\mathbf{M}$ is rank deficient if and only if its determinant is zero, i.e., $\det(M) = 0$. The determinant is a polynomial in $\{z_i : i \in [k]\}$. By induction, one can show that if a finite degree multivariate polynomial is not identically zero, the set of roots has zero Lebesgue measure (for example, see [19]). The uniform measure over the simplex is absolutely continuous with respect to Lebesgue measure. Hence, the set of roots of a finite degree multivariate polynomial has measure zero with respect to the uniform measure over the simplex.

To show that $\det(\mathbf{M})$ is not identically zero, it is sufficient to choose a set of $z_i's$ for which determinant is non-zero. First, observe that by Corollary 6, each matrix

$$\begin{bmatrix} x_{i,1}\mathbf{y}_{i,1} \\ x_{i,2}\mathbf{y}_{i,2} \\ \vdots \\ x_{i,m}\mathbf{y}_{i,m} \end{bmatrix} \tag{B.31}$$

is full rank with probability 1. Let $z_1 = 1$ and $z_j, \forall j \in \{2, 3, \ldots, k\}$. Then $\det(\mathbf{M}) \neq 0$ since $\mathbf{M}$ is full rank. Therefore, the determinant, which is a polynomial in $\{z_i : i \in [k]\}$ is not identically zero. This concludes the proof that with probability 1, $\text{rank}(\mathbf{M}) = n > k$. □

B.5 Comparing LatentSearch with EM, NMF and Gradient descent

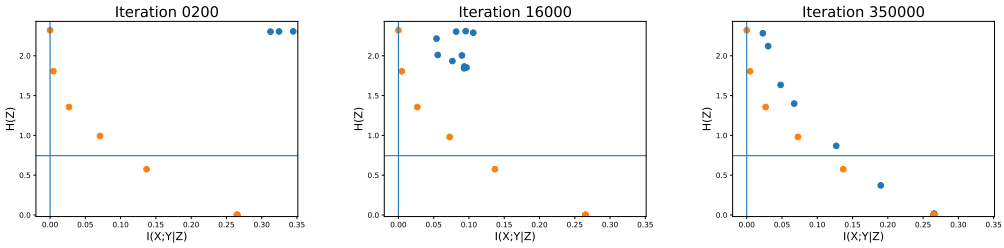
B.5.1 Comparison to gradient descent

We observed that iterative update step is slightly faster than the gradient descent step: Average time for iterative update:0.000063 seconds. Average time for gradient update: 0.000078 seconds. More importantly, gradient descent

takes much longer to converge and does not even achieve the same performance.

As observed in Figure B.2, gradient descent converges only after 350000 iterations, whereas we observed that iterative update converges after around 200 iterations. Based on the average update times, this corresponds to a staggering difference of 0.01 seconds for the iterative algorithm vs. 27.3 seconds for the gradient descent algorithm. Although these results are for when $n = m = k = 5$ states, we observed single iterative update to be faster than single gradient update, giving similar performance comparison results for $n = m = k = 80$ states.

The above result is for a constant step size of 0.001. With smaller step size, convergence slows down even further. With larger step size, gradient descent does not converge.



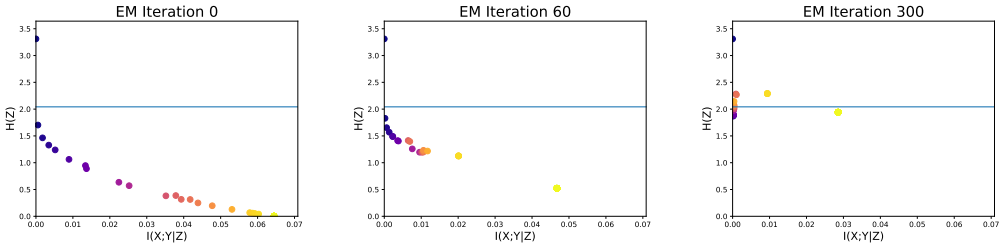
(a) (b) (c)

Figure B.2: Comparison of the iterative algorithm with gradient descent. Blue points show the trajectory of gradient descent, whereas orange points show the trajectory for Algorithm 2 for 10 randomly initialized points with different β values in loss (3.1). Gradient descent takes 350,000 iterations to converge whereas iterative algorithm converges in about 200 iterations. Moreover, the points achieved by iterative algorithm are strictly better than gradient descent after convergence.

B.5.2 Comparison with EM algorithm

EM is the first algorithm suggested for solving the pLSA problem [53]. For the details of EM within this framework, please see [53]. However the EM algorithm for pLSA problem does not have any incentive to minimize the entropy of the latent factor.

In order to see how EM affects the entropy of the discovered latent variable, we run EM algorithm by initializing it at the points that are output by LatentSearch (Algorithm 2). Results are illustrated in Figure B.3. We observe that the points obtained in the $I - H$ plane migrate towards $I(X;Y|Z) = 0$ line, while staying above what we believe is a fundamental lower bound curve. We have not observed any improvement to our algorithm by this additional step, as it leads to increased entropy latent variables.

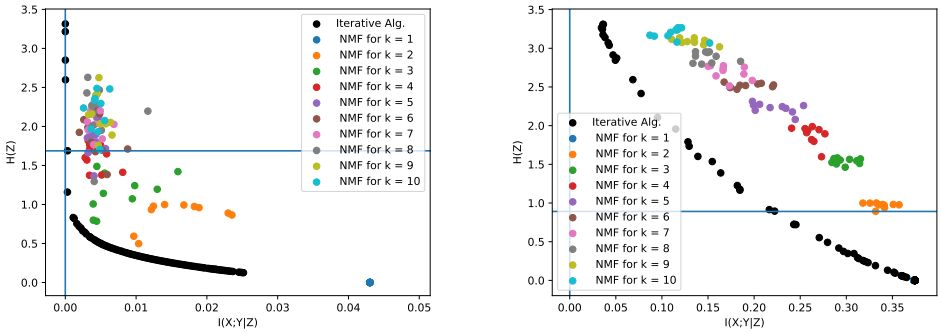


(a) (b) (c)

Figure B.3: Applying EM to the output of iterative algorithm migrates points to $I(X;Y|Z = 0)$ line: (a) Latent variables discovered by LatentSearch (Algorithm 2) shown on the $I(X;Y|Z) - H(Z)$ plane. (b,c) After applying EM algorithm on the points in (a) after 60 and 300 iterations. Observe that the points always remain above the line depicted by LatentSearch (Algorithm 2).

B.5.3 Comparison with NMF

Consider the joint distribution matrix $\mathbf{M}$. Suppose we find an approximation to this matrix as $\mathbf{M} \approx \mathbf{UV}$ where the common dimension of $\mathbf{U}, \mathbf{V}$ is k through NMF. This is equivalent to setting the dimension of the latent variable to k. This can be seen as a hard entropy threshold on the entropy of the latent factor since $H(Z) \leq \log(k)$. We can sweep through different dimensions and see how NMF performs compared to LatentSearch (Algorithm 2). Note that NMF is in general hard to solve. A commonly used approach is the iterative algorithm: Initialize $\mathbf{U}_0, \mathbf{V}_0$. Find the best $\mathbf{U}_1$ such that $\mathbf{M} \approx \mathbf{U}_1\mathbf{V}_0$. Then find the best $\mathbf{V}_1$ such that $\mathbf{M} \approx \mathbf{U}_1\mathbf{V}_1$ and iterate. In the experiments, we used this iterative algorithm together with l_1 loss.



(a) Causal Graph $X \leftarrow Z \rightarrow Y$ (b) Causal Graph $X \leftarrow Z \rightarrow Y, X \rightarrow Y$

Figure B.4: Comparison of the iterative algorithm to NMF for when $|X| = |Y| = 20, |Z| = 10$. When the true model comes from the causal graph $X \leftarrow Z \rightarrow Y$ in (a), iterative algorithm successfully finds latent variables that with entropy at most true latent entropy (shown as blue horizontal line), whereas NMF cannot achieve the same performance, irrespective of the dimension restriction to the latent variable. In (b) data comes from the causal model $X \leftarrow Z \rightarrow Y, X \rightarrow Y$. Although neither algorithm can identify a latent factor that makes X, Y conditionally independent (vertical blue line), iterative algorithm finds strictly better latent factors in terms of both small entropy and conditional mutual information between X, Y.

Appendix C

Appendix for Cost Optimal Intervention Design

C.1 Proof of Theorem 7

One direction is trivial: Consider a $(G, \mathcal{C})$ separating system. For every edge there is an intervention where only one endpoint is intervened. This edge is in the cut and learned. Constraints are over the subsets of the graph separating system, which directly correspond to interventions. Hence interventions also obey the constraint $\mathcal{C}$. For the other direction, we use the following observation from [122], which is implicit in the proof of Theroem 6 in [122].

Lemma 20. *Let G be an undirected chordal graph. Consider any clique C of G. There is a directed graph D with skeleton G with no immoralities such that, the vertices C come before any other vertices in the partial order that determine D. If this D is the underlying causal graph, knowing the causal edges outside this clique does not help identify any edges within the clique.*

The lemma essentially states that, Meek rules do not aid in identifying the edges within a clique, if the clique vertices come before any other vertex in the partial order of the underlying causal DAG.

Assume that there is an edge that is not separated by the set of interventions. If the underlying causal DAG has partial order that starts with the

nodes at the endpoints of this edge, then knowing every other edge does not help learn the direction of this edge by Lemma 20 (notice that an edge is a clique of size 2). Thus this set of interventions cannot learn every causal graph with the given skeleton.

C.2 Proof of Theorem 8

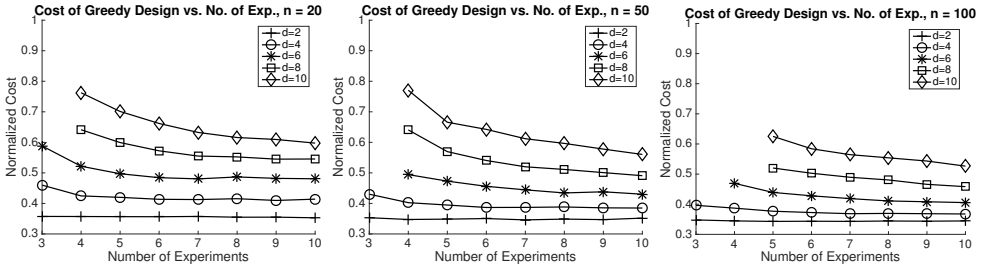
Consider the graph separating system matrix $\mathbf{M}$: Let $\mathbf{M} \in \{0,1\}^{n,m}$ be a 0-1 matrix, where $\mathbf{M}(i,:) \neq \mathbf{M}(j,:), \forall (i,j) \in E$. Since every set of interventions must be a graph separating system by Theorem 7, we can work with the corresponding graph separating system matrices. Notice that any graph separating system corresponds to some proper coloring due to 4. Thus, any set of vertices that has identical rows in $\mathbf{M}$ should be within the same color class. We know in any proper coloring, each color class is an independent set. Then, over all proper colorings, the color class with maximum weight is given by the maximum weighted independent set. Since each row of $\mathbf{M}$ is either the all-zero vector, or contains at least a single 1, the total cost is minimized by assigning the all-zero vector to the vertices belonging to the maximum weighted independent set, and using distinct weight-1 vectors for the remaining rows. The induced graph on the vertices outside the maximum weighted independent set is still chordal and has the chromatic number at most χ. Thus, we need an $n \times \chi$ matrix $\mathbf{M}$, hence χ experiments in total to minimize the total intervention cost.

C.3 Proof of Theorem 9

In this section, we show that we can write the total cost of the interventions constructed by a given graph coloring can be written as a linear objective in terms of $x_{i,k}$.

First, we illustrate the cost incurred by a given separating system. Consider the color separating system in Figure 4.1b. Notice that the rows of M that correspond to vertices within a fixed color class are the same. For example $S = \{U2, U4\}$ is a color class, and both rows are $[0, 1]$. Recall that the columns where a particular row is 1 indicate the interventions which contain that variable. The cost incurred by any vertex is the number of times the vertex is intervened on times the cost of intervening on that vertex. The cost incurred by a set of vertices is the sum of the cost incurred by each vertex within the set. Vertices within a color class are intervened on the same number of times since they have the same rows in the separating system matrix $\mathbf{M}$. Thus, the cost of a color class S is given by $cost(S) = |r_S|_1 \sum_{i \in S} w_i$, where r_S is the row of any node from color class S in M, and $|r_S|_1$ is the number of 1s in r_S.

Notice that the exact labeling of rows do not matter for the separating system: We only need vertices with different colors to correspond to different rows. Since the cost of a color class is proportional to the number of 1s in its row vector, an optimum graph separating system given a coloring should assing vectors with smaller weight if possible, in order to minimize the total cost. Hence, in Figure 4.1b, instead of assigning $[1, 1]$ as the characteristic vector of



(a) n=20 (b) n=50 (c) n=100

Figure C.1: Uniform weights $w_i \sim \mathcal{U}[0, 2]$. n: no. of vertices, d: Sparsity parameter of the chordal graph. Each datapoint is the average cost incurred by the greedy intervention design over 1000 randomly sampled causal graphs for a given number of experiments. The expected average cost of all the edges is $\mathbb{E}[w_i] = 1$. The cost incurred by the intervention design is normalized by n. As observed, the cost incurred increases gradually as the number of experiments are reduced, or graph becomes denser. For sparse graphs, proposed construction incurs low cost even for up to 3 experiments.

S, we can assign $[0, 1]$ without affecting the separating system property. Since 3 colors are sufficient, we do not need to use $[1, 1]$ vector.

In general, given a number of interventions m, we need to construct a set of coloring labels to assign to each color. Suppose the causal graph has n variables. If $m \leq \log n$, then every length-m binary vector should be available, since the number of colors can be up to n. If $m > \log n$, using all labels give more number of colors than we can use to search over all proper colorings. Hence, in this case, we choose the labels with smallest weight until we find n coloring labels. This ensures that the integer programming formulation does not have exponentially many variables, even when number of interventions is allowed to be n. Thus we construct a $\mathbf{b}$ vector, to be used as the weight of

color labels as follows:

$$\mathbf{b} = [0, 1, 1, \ldots, 1, 2, 2, \ldots, 2, 3, 3, \ldots, p, p, \ldots, p], \tag{C.1}$$

where p is such that $\sum_{i=0}^{p-1} \binom{m}{i} < \min(2^m, n)$ and $\sum_{i=0}^{p} \binom{m}{i} \geq \min(2^m, n)$. i appears $\binom{m}{i}$ times if $i < p$ and $\min 2^m, n - \sum_{j=0}^{p-1} \binom{m}{j}$ times if $i = p$. For notational convenience, let $t \coloneqq \min(2^m, n)$.

Standard coloring formulation assigns a variable $x_{i,j}$ to every node i and color j: $x_{i,j} = 1$ if node i is colored with color j, and 0 otherwise. Each vertex is assigned a single color. Every pair of adjacent vertices are assigned different colors, which can be captured by $x_{i,k} + x_{j,k} \leq 1, \forall (i,j) \in E, \forall k \in [t]$. Then, using this standard coloring formulation, we can write our optimization problem as follows:

$$\begin{aligned} \min \quad & \sum_{j=0}^{t} \sum_{i=1}^{n} w_i x_{i,j} \mathbf{b}(j) \\ \text{s. t.} \quad & \sum_{j=1}^{m} x_{i,j} = 1, \forall i \in [n] \\ & x_{i,k} + x_{j,k} \leq 1 \forall (i,j) \in E, \forall k \in [t] \\ & x_{i,j} \in \{0, 1\} \end{aligned} \tag{C.2}$$

C.4 Uniquely Colorable Graphs

Next, we give a special case, which admits a simple solution without restricting the graph class. Suppose G is uniquely 2^m- colorable, where m is the maximum number of interventions we are allowed to use. Then there is

only a single coloring up to permutations of colors. Hence the costs of color classes are fixed. Now we can simply sort the color classes in the order of decreasing cost, and assign row vectors of $\mathbf{M}$ to these color classes in the order of increasing number of 1s. This assures that the total cost of interventions is minimized.

C.5 Implementation Details

First, we need to define a perfect elimination ordering:

Definition 16. *A perfect elimination ordering (PEO) $\sigma_p = \{v_1, v_2 \dots v_n\}$ on the vertices of an undirected chordal graph G is such that for all i, the induced neighborhood of v_i on the subgraph formed by $\{v_1, v_2 \dots v_{i-1}\}$ is a clique.*

It is known that an undirected graph is chordal if and only if it has a perfect elimination ordering. We use this fact to generate chordal graphs based on a randomly chosen perfect elimination ordering: First we choose a random permutation to be the perfect elimination ordering for the chordal graph. Then the i^{th} vertex is connected to each node in $S_i = \{j : j < i \text{ with respect to PEO}\}$ with respect to the PEO independently randomly with probability $\left(\frac{d}{i}\right)^{2/3}$. A random vertex from S_i is chosen to be a parent of i with probability 1 to keep the graph connected. The parent set are connected to each other to assure the ordering is a PEO.

C.6 Frank's Algorithm

Consider a PEO $\sigma = \{v_n, v_{n-1}, \ldots, v_1\}$. At step i, skip the vertex v_i if it has weight $w_i = 0$. Otherwise, mark it red and reduce the weight of all its neighbors that are before v_i in the PEO by w_i, and set $w_i = 0$. After n steps, we have a set of vertices colored red. Parse this set in the order of σ and convert a red vertex to blue if it does not have any neighbor $j < i$ in σ which is already colored blue. [39] proves that this algorithm outputs the maximum weighted independent set.

C.7 Additional Simulations

In this section we provide additional simulations for when the graph weights are uniformly distributed $w_i \sim \mathcal{U}[0, 2]$. The results are given in Figure 4.2. Similar to the exponentially distributed weigths, the main factor determining the cost is the graph spartiy, which is captured by parameter d.

Appendix D

Appendix for Experimental Design for Learning Causal Graphs with Latent Variables

D.1 Blocking and Non-Blocking paths in DAGs

Consider any directed acyclic graph $D = (V, E)$. A path P from node v_0 to v_{k+1} is a sequence of nodes $P = v_0, v_1, v_2 \dots v_k, v_{k+1}$ such that for all $i \in [0 : k]$, either (v_i, v_{i+1}) or (v_{i+1}, v_i) is a directed edge in E. A node v_i with respect to a path P is said to be a collider if (v_{i+1}, v_i) and (v_{i-1}, v_i) exists in E, i.e. two directed edges collide at v_i. A path P between nodes a and b is said to be non-blocking with respect to a set S if and only if for every collider v on P, either v in S or a descendant of v is in S and no non collider v in P is in S. S is said to block a path P if it is not non-blocking.

D.2 Proof of Lemma 5

First, we prove the forward direction. Suppose that X_i and X_j are dependent under the post-interventional causal model $\mathcal{M}_S$. By the assumption of post-interventional faithfulness, this implies that there is a non-blocking path P between X_i and X_j in $D_\ell[S]$. Suppose V in P is a collider because either that or one of its descendant has to be in the conditioning set. Since,

there is no conditioning set and we are testing only marginal dependence, there cannot be any collider V in P. Also note that there are no incoming edges into X_i in either $D_\ell[S]$ or $D[S]$ because there X_i is in the set S intervened on. Therefore, there cannot be any internal node $V \neq X_i, X_j$ in P with no incoming arrows because then either the path must have a collider or X_i must have incoming arrows. Since both these events are ruled out, no internal node V can have in-degree 0 or 2. The only option is for it to be a directed path from X_i to X_j in $D_\ell[S]$. This implies that no latent variable in $\mathcal{L}$ is a part of the path since latents have 0 in-degree. This implies that P is a directed path from X_i to X_j in $D[S]$ also. This proves one direction.

For the other direction, suppose there is a directed path from X_i to X_j in $D[S]$, it is still a directed path from X_i to X_j in $D_\ell[S]$ as no latents are involved. This implies that it is a non-blocking path between X_i to X_j. By the post-interventional faithfulness assumption, this implies that X_i and X_j are dependent in the causal model $\mathcal{M}_S$. This completes the proof in the other direction. □

D.3 Proof of Lemma 6

Consider the $\lceil \log n \rceil$ length binary expansions of numbers from $1 : n$. For every bit i, create a set S_i with the numbers where the i-th digit is 1 and another set S_i' with the numbers where the i-th digit is 0. The family of sets $\{S_i, S_i'\}$ if a strong separating system. It is easy to check the condition.

D.4 Proof of Theorem 10

It is enough to show that every directed edge e in the observable graph D is included at some step in E. Let the directed edge e be from U to V in the observable graph. Due to the strong separating system property, there is one intervention set S such that $U \in S,\ V \notin S$. Therefore, in that post interventional graph $D[S]$, U is an ancestor of V and therefore by Lemma 5, it is included in E after processing Line 9 for S. This implies that all directed edges of D (in addition to other ancestral relationships) are included in E. Therefore, the transitive closure at the end yields D_{tc}. □

D.5 Proof of Lemma 7

Consider the pair (X_i, X_j), where $X_i \notin S, X_j \in S, X_j \notin Pa_i$. In the post-interventional graph, X_j has no parents, including the possible latent variables. Any d-connecting path from X_j to X_i must end with an incoming arrow at X_i since any path that ends with an outgoing arrow at X_i is closed as it travels through a collider and all colliders are closed since no variable is conditioned in the graph. Thus any X_j not in the parent set of X_i is independent from X_i. Any parent of X_i will clearly be statistically dependent with X_i. □

D.6 Proof of Lemma 9

It is easy to show that $\text{Tr}((D[S])_{\text{tc}}) = \text{Tr}(D[S])$ from the properties of $\text{Tr}(\cdot)$. We will prove the rest of the implication by contradiction. Suppose (Y, V_i) is not a directed edge in $\text{Tr}(D[S])$, then the ancestral relation (Y, V_i) needs to be accounted for by another directed path from Y to V_i in $\text{Tr}(D[S])$. This is due to the definition of transitive reduction of $D[S]$ and the fact that V_i is connected to all its direct parents in the post interventional graph as V_i has not been intervened on. This implies that there is a directed path starting from Y and ending at some other parent $X \neq Y$ of V_i in $D[S]$. This implies that such a direct parent X has an incoming edge. This cannot happen since then by the partial ordering $\pi(\cdot)$, all direct parents of V_i above Y in the partial order have been intervened on and thereby leaving no incoming edges onto those nodes in $D[S]$. This implies a contradiction. This implies that the directed edge (Y, V_i) is present in $\text{Tr}(D[S])$.

D.7 Proof of Theorem 12

Consider a directed edge (Y, V_i) in D. Let the number of direct parents of V_i above Y in the partial order be d_i. Clearly, $d_i \leq d_{\max}$. Observe that in one run of the inner for loop at Line 4, the probability V_i is excluded from S and that all direct parents of V_i above Y in the partial order are included in S

is given by:

$$\begin{aligned}\Pr\left(V_i \notin S \cap \{X : \pi(X) \geq \pi(Y),\ (X, V_i) \in D\} \subseteq S\right) &= \frac{1}{d_{\max}}\left(1 - 1/d_{\max}\right)^{d_i} \\ &\geq \frac{1}{d_{\max}}\left(1 - 1/d_{\max}\right)^{d_{\max}} \\ &\overset{a}{\geq} \frac{1}{d_{\max}}\frac{1}{4} \qquad \text{(D.1)}\end{aligned}$$

(a)- This is because $\frac{1}{4} \leq (1 - 1/n)^n \leq \frac{1}{e},\ \forall n \geq 2$ and $d_{\max} \geq 2$. Here, e is the base of the natural logarithm. Let $\mathcal{A}_i(Y)$ be the event: $V_i \notin S \cap \{X : \pi(X) \geq \pi(Y),\ (X, V_i) \in D\} \subseteq S$. By Lemma 9, the event $\mathcal{A}_i(Y)$ implies that the directed edge (Y, V_i) is included in the output and the output cannot contain any extra edges as edges set of $Tr(D[S])$ is contained in $D[S]$ which is contained in D. Now, in over $4cd_{\max}\log n$ runs of the outer for loop we upper bound the probability of failure, i.e. $(\mathcal{A}_i(Y))^c$ is true over all runs of the outer for loop.

$$\Pr((\mathcal{A}_i(Y))^c,\ \text{for all runs}) \leq (1 - \frac{1}{4d_{\max}})^{4cd_{\max}\log n} \leq \exp(-c\log n) \leq \frac{1}{n^c}. \qquad \text{(D.2)}$$

Union bounding over all possible bad events for every pair Y, V_i in the graph D, the probability of failure is at most $\frac{1}{n^{c-2}}$.

D.8 Proof of Theorem 13

Under the interventional causal faithfulness assumption, we only need to show that, under the intervention $(Pa_i \cup Pa_j)$, two non-adjacent nodes V_i, V_j will be d-separated if and only if there is no latent variable that causes both.

Consider any undirected path (a path that does not necessarily respect edge directions) between V_i and V_j in the post-interventional graph. For convenience, we say the path starts at V_i and ends at V_j without loss of generality. Since the observable parents are intervened on, any d-connecting path must start with either a child of V_i or a latent parent of V_i and end with either a child of V_j or a latent parent of V_j. If the path starts and ends with the children of V_i and V_j, then there must be a collider on the path, which closes the path since no variable is conditioned on in the graph. Consider a path that starts with a latent parent of V_i and ends with a latent parent of V_j. Since latent variables are non-adjacent, these latent variables can only be connected through their children. Hence, by the same argument that any path through the children of two variables must have a collider, the path between these two latent variables is closed, making the path between V_i and V_j closed. Consider a path that starts with a latent parent of V_i and ends with a child of V_j. The path should arrive at the child of V_j through one of its parents, as otherwise there will be a collider on the path by the same argument above. But then the child of V_j is a collider on this path, making the path closed. Hence, any path between V_i and V_j in the post-interventional graph is closed. This proves the first part of the Theorem.

Now, we show that the set S can actually larger without affecting anything. An intervention can only affect the descendant variables in the causal graph, since all the backdoor paths are closed. In the post-interventional graph under (Pa_i, Pa_j), X_i and X_j do not have any ancestors other than the direct

parents Pa_i and Pa_j. Hence, intervening on the variables in $S \backslash (Pa_i \cup Pa_j)$ does not affect the interventional distribution between X_i, X_j under $do(Pa_i, Pa_j)$.

D.9 Proof of Lemma 10

Consider the undirected version G of D. Consider the complement graph G^c. A set of independent sets $\mathcal{I} = \{I_1, I_2, \ldots, I_m\}$ in G that cover every non-edge in G is an edge-clique cover in the complement graph G^c. The minimum edge-clique cover is also known as the intersection number of the graph. When D has degree d, G^c has degree at least $n - d$. It was shown in [3] that the intersection number of graphs with degree at least $n - d$ is at most $2e^2(d+1)^2 \ln(n)$ by a probabilistic method argument that employs a randomized algorithm as follows: Choose every vertex independently with probability $\frac{1}{d+1}$ into a set S. Then prune S to delete vertices that are not connected to the rest of the vertices in S in G^c to obtain a clique S' in G^c. Repeat this $4e^2(d+1)^2 \ln(n)$ times to generate many cliques. By repeating the calculations in [3], it can be easily shown that the above randomized procedure succeeds with probability at least $1 - \frac{1}{n^2}$ in returning an edge clique cover of G^c.

D.10 Proof of Theorem 15

Proof that if there are no latents, then equality in Thm 15 holds. We use the notation L^i for the set of latent parents of V_i. Also, with a slight abuse of notation, we use Pa_j to refer to all observable parents of V_j

except V_i. Suppose there does not exist a variable L_k that is the parent of both V_i and V_j ($L^i \cap L^j = \emptyset$). We can write $V_j = g(V_i, Pa_j, L_j)$.

$$\Pr(V_j|do(V_i = v_i, Pa_i = pa_i, Pa_j = pa_j)) \tag{D.3}$$

$$= \sum_{l_j} \Pr(V_j|L^j = l_j, do(V_i = v_i, Pa_i = pa_i, Pa_j = pa_j)) \tag{D.4}$$
$$\Pr(L^j = l_j|do(V_i = v_i, Pa_i = pa_i, Pa_j = pa_j))$$

$$= \sum_{l_j} \Pr(V_j|L^j = l_j, do(V_i = v_i, Pa_i = pa_i, Pa_j = pa_j)) \tag{D.5}$$
$$\Pr(L^j = l_j)$$

(D.4) is obtained through conditioning and marginalizing out the latent parents of X_j. (D.5) is due to the fact L^j are non-descendants of the set $\{X_i, Pa_i, Pa_j\}$. We also have,

$$\Pr(V_j|V_i = x_i, do(Pa_i = pa_i, Pa_j = pa_j))$$

$$= \sum_{l_j} \Pr(V_j|L^j = l_j, V_i = v_i, do(Pa_i = pa_i, Pa_j = pa_j)) \tag{D.6}$$
$$\Pr(L^j = l_j|V_i = v_i, do(Pa_i = pa_i, Pa_j = pa_j))$$

$$= \sum_{l_j} \Pr(V_j|L^j = l_j, do(V_i = v_i, Pa_i = pa_i, Pa_j = pa_j)) \tag{D.7}$$
$$\Pr(L^j = l_j|V_i = v_i, do(Pa_i = pa_i, Pa_j = pa_j))$$

$$= \sum_{l_j} \Pr(V_j|L^j = l_j, do(V_i = v_i, Pa_i = pa_i, Pa_j = pa_j)) \tag{D.8}$$
$$\Pr(L^j = l_j)$$

(D.6) is obtained through conditioning and marginalizing out the other parents of Y. (D.7) is due to Lemma 21.

Lemma 21. *Let S_i, T_i be subsets of Pa_X such that $S_1 \cup S_2 = Pa_X, S_1 \cap S_2 = \emptyset$ and $T_1 \cup T_2 = Pa_X, T_1 \cap T_2 = \emptyset$. Then*

$$\Pr(X|S_1, do(S_2)) = \Pr(X|do(Pa_X)) = \Pr(X|Pa_X) = \Pr(X|T_1, do(T_2)). \tag{D.9}$$

Proof. The proof uses the invariance principle of causal Bayesian networks: The invariance principle (see Definition 1.3.1 (iii) in page 24 in ([105]) states that $\Pr(X|Pa_X = pa_X, do(Z = z)) = \Pr(X|Pa_X = pa_X)$ as long as $X \notin Z$ and $Z = z$ is consistent with $Pa_X = pa_X$. Let $Z = S_2$. Then $\Pr(X|Pa_X = pa_X, do(S_2 = s_2)) = \Pr(X|S_1 = s_1, do(S_2 = s_2))$, where $S_1 = Pa_X \backslash S_2$. Thus $\Pr(X|Pa_X = pa_x) = \Pr(X|S_1 = s_1, do(S_2 = s_2))$. From Property 1 in page 24 of [Pearl2009], we have $\Pr(X|Pa_X = pa_X) = \Pr(X|do(S_1 = s_1, S_2 = s_2))$. Choosing T_1, T_2 instead of S_1, S_2 we can show that $\Pr(X|Pa_X = pa_X) = \Pr(X|T_1 = t_1, do(T_2 = t_2))$, which completes the proof. □

(D.8) is due to the following: For L^j, we have two possibilities: (i) : X_i is a non-descendant of L^j. Then the result is implied by the Markov condition. (ii). X_i is a descendant of L^j. Then there are directed paths from L^j to X_i. Note that all these paths must go through variables in Pa_i. Then, the result follows from the fact that $L^j \perp\!\!\!\perp X_i|do(Pa_i)$.

Proof that if there are latents, then equality in Thm 15 does not hold. Let us assume that the between two variables V_i and V_j a latent L_{ij} exists. Assume V_i is a parent of V_j. Suppose in contradiction, equality in

Thm 15 holds. Then we have the following: Denote $\text{do}(Pa_i = pa_i, Pa_j = pa_j)$ by the shorthand $\text{do}(pa_{ij})$. Latent variable L_{ij} influences V_i and V_j and U_i is the exogenous variable tied to V_i. All other latents are denoted by L's and exogenous variables by U's. Consider the set of latents $\mathbf{L}_i$ which are related to V_i and let $\mathbf{l}_i$ be the values they take.

$$
\begin{aligned}
&\Pr(V_j|V_i = v_i, do(Pa_i = pa_i, Pa_j = pa_j)) = \\
&\Pr(V_j|do(V_i = v_i), do(Pa_i = pa_i, Pa_j = pa_j)) \\
&\Rightarrow \sum_{\{L_p=l_p, U_q=u_q\}_{p,q}} \Pr(V_j|V_i = v_i, \mathrm{do}(pa_{ij}), \{u_q, l_p\}) \\
&\qquad\qquad \Pr(\{u_q, l_p\}|V_i = v_i, \mathrm{do}(pa_{ij})) = \\
&\sum_{\{L_p=l_p, U_q=u_q\}_{p,q}} \Pr(V_j|\mathrm{do}(V_i = v_i), \mathrm{do}(pa_{ij}), \{u_q, l_p\}) \\
&\qquad\qquad \Pr(\{u_q, l_p\}|\mathrm{do}(V_i = v_i), \mathrm{do}(pa_{ij})) \\
&\overset{a}{\Rightarrow} \sum_{\{L_p=l_p, U_q=u_q\}_{p,q}} \Pr(V_j|\mathrm{do}(V_i = v_i), \mathrm{do}(pa_{ij}), \{u_q, l_p\}) \\
&\qquad \Pr(\mathbf{L}_i = \mathbf{l}_i, U_i = u_i|V_i = v_i, \mathrm{do}(pa_{ij})) \prod_{p:L_p \notin \mathbf{L}_i} \Pr(l_p) \prod_{q:U_q \neq U_i} \Pr(u_q) \\
&= \sum_{\{L_p=l_p, U_q=u_q\}_{p,q}} \Pr(V_j|\mathrm{do}(V_i = v_i), \mathrm{do}(pa_{ij}), \{u_q, l_p\}) \\
&\qquad \Pr(\mathbf{L}_i = \mathbf{l}_i, U_i = u_i) \prod_{p:L_p \notin \mathbf{L}_i} \Pr(l_p) \prod_{q:U_q \neq U_i} \Pr(u_q) \\
&\Rightarrow \sum_{\{L_p=l_p, U_q=u_q\}_{p,q}} \Pr(V_j|\mathrm{do}(V_i = v_i), \mathrm{do}(pa_{ij}), \{u_q, l_p\}) \Pr(\mathbf{L}_i = \mathbf{l}_i, U_i = u_i) \\
&\frac{\Pr(V_i = v_i|\mathbf{L}_i = \mathbf{l}_i, U_i = u_i, \mathrm{do}(pa_{ij}))}{\Pr(V_i = v_i|\mathrm{do}(pa_{ij})} \prod_{p:L_p \notin \mathbf{L}_i} \Pr(l_p) \prod_{q:U_q \neq U_i} \Pr(u_q) \\
&= \sum_{\{L_p=l_p, U_q=u_q\}_{p,q}} \Pr(V_j|\mathrm{do}(V_i = v_i), \mathrm{do}(pa_{ij}), \{u_q, l_p\}) \\
&\ldots \Pr(\mathbf{L}_i = \mathbf{l}_i, U_i = u_i) \prod_{p:L_p \notin \mathbf{L}_i} \Pr(l_p) \prod_{q:U_q \neq U_i} \Pr(u_q)
\end{aligned}
\tag{D.10}
$$

(a)- Once all hidden variables l_q, u_q(exogenous and latents are conditioned), then do operations and conditioning are identical. The distributions of latents

is unaffected by interventions on observables. And latents and observables are independent. When hidden variables are conditioned on v_i and its parents are intervened on all the latents that are not related to v_i.

It seems like in both sides the ratio $\frac{\Pr(V_i=v_i|\mathbf{L}_i=\mathbf{l}_i,U_i=u_i,\text{do}(pa_{ij}))}{\Pr(V_i=v_i|\text{do}(pa_{ij})}$ appears which is a function of $\mathbf{l}_i, u_i$. For most functions (parameters) in the SCM, the ratios will be different from 1 and only with measure zero over the parameter space will equality hold despite the ratio being different. This gives rise to a contradiction.

D.11 Proof of Theorem 16

In this proof, when we refer to parents we refer to parent nodes from the observable graph only. Consider a color class j resulting from the strong-edge coloring of the observable graph D. Consider one directed edge (V_t, V_h) belonging to the color class j. V_h is the vertex at the head of the edge while V_t is the vertex at the tail of the edge. First observe that $P(V_h|\text{do}(T_j, P_j)) = P(V_h|\text{do}(Pa_t, Pa_h, V_t))$. Here, Pa_h is the set of parent nodes of V_h not including V_t. Pa_t is the set of parent nodes of V_t. This is because V_h has no other parent other than V_t inside the color class due to the strong edge coloring property. Note that $V_t \in T_j$. Therefore, once all parents of V_h are intervened on, other interventions in the graph do not make any difference on the computation of $P(V_h|\text{do}(Pa_t, Pa_h, V_t))$. Since there is no conditioning involved, Latents do not affect the equality.

Now, observe that $P(V_h|V_t, \text{do}(Pa_t, Pa_h)) = P(V_h|V_t, \text{do}(P_j))$. This is

because all parents of V_h except V_t are in P_j due to the strong edge coloring property. All parents of V_t are in P_j due to the strong edge coloring property. Since parents of V_t are intervened on, the random variable V_t is independent of any other intervened variable in the system. Further, the joint distribution of (V_h, V_t) is invariant to conditioning on other intervened variables in the system as all parents of V_h, V_t have been intervened on (except the parent V_t of V_h). Intervention on non-parents of either V_t and V_h have no effect on the joint distribution of (V_h, V_t) on the post-interventional graph where Pa_h, Pa_t have been intervened on. Therefore, all quantities required for the do and see test can be calculated from just two interventions in the algorithm. This is true within a color class. This proves the theorem. The experimental budget is quite obvious from the algorithm. □

Appendix E

Appendix for CausalGAN

E.1 Causality Background

Formally, a structural causal model is a tuple $\mathcal{M} = (\mathcal{V}, \mathcal{E}, \mathcal{F}, \mathbb{P}_E(.))$ that contains a set of functions $\mathcal{F} = \{f_1, f_2, \ldots, f_n\}$, a set of random variables $V = \{X_1, X_2, \ldots, X_n\}$, a set of exogenous random variables $\mathcal{E} = \{E_1, E_2, \ldots, E_n\}$, and a probability distribution over the exogenous variables $\mathbb{P}_\mathcal{E}$[1]. The set of observable variables $\mathcal{V}$ has a joint distribution implied by the distributions of $\mathcal{E}$, and the functional relations $\mathcal{F}$. This distribution is the projection of $\mathbb{P}_\mathcal{E}$ onto the set of variables $\mathcal{V}$ and is shown by $\mathbb{P}_\mathcal{V}$. The causal graph D is then the directed acyclic graph on the nodes $\mathcal{V}$, such that a node X_j is a parent of node X_i if and only if X_j is in the domain of f_i, i.e., $X_i = f_i(X_j, S, E_i)$, for some $S \subset V$. The set of parents of variable X_i is shown by Pa_i. D is then a Bayesian network for the induced joint probability distribution over the observable variables $\mathcal{V}$. We assume causal sufficiency: Every exogenous variable is a direct parent of at most one observable variable.

[1]The definition provided here assumes causal sufficiency, i.e., there are no exogenous variables that affect more than one observable variable. Under causal sufficiency, Pearl's model assumes that the distribution over the exogenous variables is a product distribution, i.e., exogenous variables are mutually independent.

E.2 Proof of Proposition 4

Note that D_1 and D_2 are the same causal Bayesian networks [105]. Under the causal sufficiency assumption, interventional distributions for causal Bayesian networks can be directly calculated from the conditional probabilities and the causal graph. Thus, $\mathcal{M}_1$ and $\mathcal{M}_2$ have the same interventional distributions. □

E.3 Helper Lemmas for CausalGAN

In this section we use $\mathbb{P}_r(l, x)$ for the joint data distribution over a single binary label l and the image x. We use $\mathbb{P}_g(l, x)$ for the joint distribution over the binary label l fed to the generator and the image x produced by the generator. Later in Theorem 19, l is generalized to be a vector.

The following restates Proposition 1 from [42] as it applies to our discriminator:

Proposition 7 ([42])**.** *For fixed G, the optimal discriminator D is given by*

$$D^*_G(x) = \frac{\mathbb{P}_r(x)}{\mathbb{P}_r(x) + \mathbb{P}_g(x)}. \tag{E.1}$$

Second, we identify the optimal Labeler and Anti-Labeler. We have the following lemma:

Lemma 22. *The optimum Labeler has $D_{LR}(x) = \mathbb{P}_r(l = 1|x)$.*

Proof. The proof follows the same lines as in the proof for the optimal discriminator. Consider the objective

$$\begin{aligned}&\rho\mathbb{E}_{x\sim\mathbb{P}_r(x|l=1)}\left[\log(D_{LR}(x))\right]+(1-\rho)\mathbb{E}_{x\sim\mathbb{P}_r(x|l=0)}\left[\log(1-D_{LR}(x)\right]\\&=\int\rho\mathbb{P}_r(x|l=1)\log(D_{LR}(x))+(1-\rho)\mathbb{P}_r(x|l=0)\log(1-D_{LR}(x))dx\end{aligned}\tag{E.2}$$

Since $0 < D_{LR} < 1$, D_{LR} that maximizes (6.3) is given by

$$D^*_{LR}(x)=\frac{\rho\mathbb{P}_r(x|l=1)}{\mathbb{P}_r(x|l=1)\rho+\mathbb{P}_r(x|l=0)(1-\rho)}=\frac{\rho\mathbb{P}_r(x|l=1)}{\mathbb{P}_r(x)}=\mathbb{P}_r(l=1|x)\tag{E.3}$$

□ □

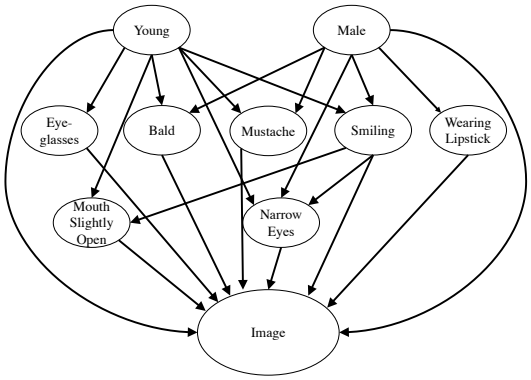


Figure E.1: The causal graph used for simulations for both CausalGAN and CausalBEGAN, called CelebA Causal Graph (G1). We also add edges (see Appendix Section E.10) to form the complete graph "cG1". We also make use of the graph rcG1, which is obtained by reversing the direction of every edge in cG1.

Similarly, we have the corresponding lemma for Anti-Labeler:

Lemma 23. *For a fixed generator with $x \sim \mathbb{P}_g(x)$, the optimum Anti-Labeler has $D_{LG}(x) = \mathbb{P}_g(l = 1|x)$.*

Proof. Proof is the same as the proof of Lemma 22. □

E.4 Proof of Theorem 17

Theorem 1.

Define $C(G)$ as the generator loss for when discriminator, Labeler and Anti-Labeler are at their optimum. Assume $\mathbb{P}_g(l) = \mathbb{P}_r(l)$, i.e., the Causal Controller samples from the true label distribution. Then the global minimum of the virtual training criterion $C(G)$ is achieved if and only if $\mathbb{P}_g(l, x) = \mathbb{P}_r(l, x)$, i.e., if and only if given a label l, generator output $G(z, l)$ has the same distribution as the class conditional image distribution $\mathbb{P}_r(x|l)$.

Proof. For a fixed generator, the optimum Labeler D^*_{LR}, Anti-Labeler D^*_{LG}, and discriminator D^* obey the following relations by Prop 7, Lemma 22, and Lemma 23:

$$\begin{aligned}(1 - D^*(x))/D^*(x) &= \mathbb{P}_g(x)/\mathbb{P}_r(x) \\ D^*_{LR}(x) &= \mathbb{P}_r(l = 1|x) \\ D^*_{LG}(x) &= \mathbb{P}_g(l = 1|x).\end{aligned} \tag{E.4}$$

Then substitution into the generator objective in (6.5) yields

$$
\begin{aligned}
C(G) &= \mathbb{E}_{x\sim p_{\mathrm{g}}(x)}\left[\log\left(\frac{1-D^*(x)}{D^*(x)}\right)\right] \\
&\quad - \rho\mathbb{E}_{x\sim p_g^1(x)}\left[\log(D^*_{LR}(X))\right] - \bar{\rho}\mathbb{E}_{x\sim p_g^0(x)}\left[\log(1-D^*_{LR}(X))\right] \\
&\quad + \rho\mathbb{E}_{x\sim p_g^1(x)}\left[\log(D^*_{LG}(X))\right] + \bar{\rho}\mathbb{E}_{x\sim p_g^0(x)}\left[\log(1-D^*_{LG}(X))\right] \\
&= \mathbb{E}_{x\sim p_{\mathrm{g}}(x)}\left[\log\left(\frac{\mathbb{P}_g(x)}{\mathbb{P}_r(x)}\right)\right] \\
&\quad - \mathbb{E}_{(l,x)\sim\mathbb{P}_g(l,x)}\left[\log(\mathbb{P}_r(l|x))\right] + \mathbb{E}_{(l,x)\sim\mathbb{P}_g(l,x)}\left[\log(\mathbb{P}_g(l|x))\right] \qquad \text{(E.5)} \\
&= \mathbb{E}_{(l,x)\sim\mathbb{P}_g(l,x)}\left[\log\left(\frac{\mathbb{P}_g(x)}{\mathbb{P}_r(x)}\right) + \log(\mathbb{P}_g(l|x)) - \log(\mathbb{P}_r(l|x))\right] \\
&= \mathbb{E}_{(l,x)\sim\mathbb{P}_g(l,x)}\left[\log\left(\frac{\mathbb{P}_g(l,x)}{\mathbb{P}_d(l,x)}\right)\right] \\
&= KL(\mathbb{P}_g \parallel \mathbb{P}_d). \qquad \text{(E.6)}
\end{aligned}
$$

where KL is the Kullback-Leibler divergence, which is minimized if and only if $\mathbb{P}_g = \mathbb{P}_d$ jointly over labels and images. (E.5) is due to the fact that $\mathbb{P}_r(l=1) = \mathbb{P}_g(l=1) = \rho$. □

E.5 Proof of Corollary 5

Corollary 1. *Suppose $C : \mathcal{Z}_1 \to \mathcal{L}$ is a causal implicit generative model for the causal graph $D = (\mathcal{V}, E)$ where $\mathcal{V}$ is the set of image labels and the observational joint distribution over these labels are strictly positive. Let $G : \mathcal{L} \times \mathcal{Z}_2 \to \mathcal{I}$ be a generator that can sample from the image distribution conditioned on the given label combination $L \in \mathcal{L}$. Then $G(C(Z_1), Z_2)$ is a causal implicit generative model for the causal graph $D' = (\mathcal{V} \cup \{Image\}, E \cup \{(V_1, Image), (V_2, Image), \ldots (V_n, Image)\})$.*

Proof. Since C is a causal implicit generative model for the causal graph D, by definition it is consistent with the causal graph D. Since in a conditional GAN, generator G is given the noise terms and the labels, it is easy to see that the concatenated generator neural network $G(C(Z_1), Z_2)$ is consistent with the causal graph D', where $D' = (\mathcal{V} \cup \{Image\}, E \cup \{(V_1, Image), (V_2, Image), \dots (V_n, Image)\})$. Assume that C and G are perfect, i.e., they sample from the true label joint distribution and conditional image distribution. Then the joint distribution over the generated labels and image is the true distribution since $\mathbb{P}(Image, Label) = \mathbb{P}(Image|Label)\mathbb{P}(Label)$. By Proposition 4, the concatenated model can sample from the true observational and interventional distributions. Hence, the concatenated model is a causal implicit generative model for graph D'. □

E.6 CausalGAN Analysis for Multiple Labels

In this section, we explain the modifications required to extend the proof to the case with multiple binary labels. The central difficulty with generalizing to a vector of labels $l = (l_j)_{1 \leq j \leq d}$ is that each labeler can only hope to learn about the posterior $\mathbb{P}(l_j|x)$ *for each* j. This is in general insufficient to characterize $\mathbb{P}_r(l|x)$ and therefore the generator can not hope to learn the correct joint distribution. We show two solutions to this problem. (1) From a theoretical (but perhaps impractical) perspective each labeler can be made to estimate the probability of each of the 2^d label combinations instead of each label. We do not adopt this in practice. (2) If in fact the label vector

is a deterministic function of the image (which seems likely for the present application), then using Labelers to estimate the probabilities of each of the d labels is sufficient to assure $\mathbb{P}_g(l_1, l_2, \ldots, l_d, x) = \mathbb{P}_r(l_1, l_2, \ldots, l_d, x)$ at the minimizer of $C(G)$. In this section, we present the extension in (1) and present the results of (2) in Section E.7.

Consider Figure 6.3 in the main text. The Labeler outputs the scalar $D_{LR}(x)$ given an image x. Previously in Section E.3 we showed that the optimum Labeler satisfies $D^*_{LR}(x) = \mathbb{P}_r(l = 1|X = x)$ for a single label. We first extend the Labeler objective as follows: Suppose we have d binary labels. Then we allow the Labeler to output a 2^d dimensional vector $D_{LR}(x)$, where $D_{LR}(x)[j]$ is the j^{th} coordinate of this vector. The Labeler then solves the following optimization problem:

$$\max_{D_{LR}} \sum_{j=1}^{2^d} \rho_j \mathbb{E}_{x \sim \mathbb{P}_r(x|l=j)} \log(D_{LR}(x)[j]), \tag{E.7}$$

where $\rho_j = \mathbb{P}_r(l = j)$. We have the following Lemma:

Lemma 24. *Consider a Labeler D_{LR} that outputs the 2^d-dimensional vector $D_{LR}(x)$ such that $\sum_{j=1}^{2^d} D_{LR}(x)[j] = 1$, where $x \sim \mathbb{P}_r(x, l)$. Then the optimum Labeler with respect to the loss in (E.7) has $D^*_{LR}(x)[j] = \mathbb{P}_r(l = j|x)$.*

Proof. Suppose $\mathbb{P}_r(l = j|x) = 0$ for a set of (label, image) combinations. Then $\mathbb{P}_r(x, l = j) = 0$, hence these label combinations do not contribute to the expectation. Thus, without loss of generality, we can consider only the combinations with strictly positive probability. We can also restrict our

attention to the functions D_{LR} that are strictly positive on these (label,image) combinations; otherwise, loss becomes infinite, and as we will show we can achieve a finite loss. Consider the vector $D_{LR}(x)$ with coordinates $D_{LR}(x)[j]$ where $j \in [2^d]$. Introduce the discrete random variable $Z_x \in [2^d]$, where $\mathbb{P}(Z_x = j) = D_{LR}(x)[j]$. The Labeler loss can be written as

$$\min -\mathbb{E}_{(x,l)\sim\mathbb{P}_r(x,l)} \log(\mathbb{P}(Z_x = j)) \tag{E.8}$$

$$= \min \mathbb{E}_{x\sim\mathbb{P}_r(x)} KL(L_x \parallel Z_x) - H(L_x), \tag{E.9}$$

where L_x is the discrete random variable such that $\mathbb{P}(L_x = j) = \mathbb{P}_r(l = j|x)$. $H(L_x)$ is the Shannon entropy of L_x, and it only depends on the data. Since KL divergence is greater than zero and $p(x)$ is always non-negative, the loss is lower bounded by $-H(L_x)$. Notice that this minimum can be achieved by satisfying $\mathbb{P}(Z_x = j) = \mathbb{P}_r(l = j|x)$. Since KL divergence is minimized if and only if the two random variables have the same distribution, this is the unique optimum, i.e., $D^*_{LR}(x)[j] = \mathbb{P}_r(l = j|x)$.

□

The lemma above simply states that the optimum Labeler network will give the posterior probability of a particular label combination, given the observed image. In practice, the constraint that the coordinates sum to 1 could be satisfied by using a softmax function in the implementation. Next, we have the corresponding loss function and lemma for the Anti-Labeler network. The

Anti-Labeler solves the following optimization problem

$$\max_{D_{LG}} \sum_{j=1}^{2^d} \rho_j \mathbb{E}_{\mathbb{P}_g(x|l=j)} \log(D_{LG}(x)[j]), \tag{E.10}$$

where $\mathbb{P}_g(x|l=j) \coloneqq \mathbb{P}(G(z,l)=x|l=j)$ and $\rho_j = \mathbb{P}(l=j)$. We have the following Lemma:

Lemma 25. *The optimum Anti-Labeler has* $D^*_{LG}(x)[j] = \mathbb{P}_g(l=j|x)$.

Proof. The proof is the same as the proof of Lemma 24, since Anti-Labeler does not have control over the joint distribution between the generated image and the labels given to the generator, and cannot optimize the conditional entropy of labels given the image under this distribution. □

For a fixed discriminator, Labeler and Anti-Labeler, the generator solves the following optimization problem:

$$\begin{aligned}
&\min_G \mathbb{E}_{x\sim p_g(x)} \left[\log \left(\frac{1-D(x)}{D(x)} \right) \right] \\
&- \sum_{j=1}^{2^d} \rho_j \mathbb{E}_{x\sim \mathbb{P}_g(x|l=j)} \left[\log(D_{LR}(X)[j]) \right] \\
&+ \sum_{j=1}^{2^d} \rho_j \mathbb{E}_{x\sim \mathbb{P}_g(x|l=j)} \left[\log(D_{LG}(X)[j]) \right].
\end{aligned} \tag{E.11}$$

We then have the following theorem along the same lines as Theorem 17 showing that the optimal generator samples from the class conditional image distributions given a particular label combination:

Theorem 19 (Theorem 1 formal for multiple binary labels)**.** *Define $C(G)$ as the generator loss as in Eqn. E.11 when discriminator, Labeler and Anti-Labeler are at their optimum. Assume $\mathbb{P}_g(l) = \mathbb{P}_r(l)$, i.e., the Causal Controller samples from the true joint label distribution. The global minimum of the virtual training criterion $C(G)$ is achieved if and only if $\mathbb{P}_g(l, x) = \mathbb{P}_r(l, x)$ for the vector of labels $l = \{l_i\}_{1 \leq i \leq 2^d}$.*

Proof. For a fixed generator, the optimum Labeler D^*_{LR}, Anti-Labeler D^*_{LG}, and discriminator D^* obey the following relations by Prop 7, Lemma 24, and Lemma 25:

$$\begin{aligned}
&(1 - D^*(x))/D^*(x) = \mathbb{P}_g(x)/\mathbb{P}_r(x) \\
&D^*_{LR}(x)[j] = \mathbb{P}_r(l = j|x) \ \forall j \\
&D^*_{LG}(x)[j] = \mathbb{P}_g(l = 1|x) \ \forall j.
\end{aligned} \tag{E.12}$$

Then substitution into the generator objective $C(G)$ yields

$$
\begin{aligned}
C(G) = & \qquad \text{(E.13)} \\
& \sum_{j=1}^{2^d} \rho_j \mathbb{E}_{x \sim \mathbb{P}_g(x|l=j)} \left[\log \left(\frac{\mathbb{P}_g(x)}{\mathbb{P}_r(x)} \right) + \log(\mathbb{P}_g(l=j|x)) - \log(\mathbb{P}_r(l=j|x)) \right] \\
& = \sum_{j=1}^{2^d} \rho_j \mathbb{E}_{x \sim \mathbb{P}_g(x|l=j)} \left[\log \left(\frac{\mathbb{P}_g(l=j,x)}{\mathbb{P}_r(l=j,x)} \right) \right] \\
& = \mathbb{E}_{(l,x) \sim \mathbb{P}_g(l,x)} \left[\log \left(\frac{\mathbb{P}_g(l,x)}{\mathbb{P}_d(l,x)} \right) \right] \\
& = KL(\mathbb{P}_g \parallel \mathbb{P}_d).
\end{aligned}
$$

where KL is the Kullback-Leibler divergence, which is minimized if and only if $\mathbb{P}_g = \mathbb{P}_d$ jointly over labels and images.

□

E.7 CausalGAN Extension to $d$abels Under Deterministic Labels

While the previous section showed how to ensure $\mathbb{P}_g(l,x) = \mathbb{P}_r(l,x)$ by relabeling combinations of a d binary labels as a 2^d label, this may be difficult in practice for a large number of labels and we do not adopt this approach in practice.

Instead, in this section, we provide the theoretical guarantees for the implemented CausalGAN architecture with d labels under the assumption that the relationship between the image and its labels is deterministic in the dataset, i.e., there is a deterministic function that maps an image to the corresponding

label vector. Later we show that this assumption is sufficient to gaurantee that the global optimal generator samples from the class conditional distributions.

First, let us restate the loss functions more formally. Note that $D_{LR}(x)$ and $D_{LG}(x)$ are $d-$dimensional vectors. The Labeler solves the following optimization problem:

$$\max_{D_{LR}} \rho_j \mathbb{E}_{x\sim\mathbb{P}_r(x|l_j=1)} \log(D_{LR}(x)[j]) + (1-\rho_j)\mathbb{E}_{x\sim\mathbb{P}_r(x|l_j=0)} \log(1-D_{LR}(x)[j]). \tag{E.14}$$

where $\mathbb{P}_r(x|l_j=0) \coloneqq \mathbb{P}(X=x|l_j=0)$, $\mathbb{P}_r(x|l_j=0) \coloneqq \mathbb{P}(X=x|l_j=0)$ and $\rho_j = \mathbb{P}(l_j=1)$. For a fixed generator, the Anti-Labeler solves the following optimization problem:

$$\max_{D_{LG}} \rho_j \mathbb{E}_{\mathbb{P}_g(x|l_j=1)} \log(D_{LG}(x)[j]) + (1-\rho_j)\mathbb{E}_{\mathbb{P}_g(x|l_j=0)} \log(1-D_{LG}(x)[j]), \tag{E.15}$$

where $\mathbb{P}_g(x|l_j=0) \coloneqq \mathbb{P}_g(x|l_j=0)$, $\mathbb{P}_g(x|l_j=0) \coloneqq \mathbb{P}_g(x|l_j=0)$. For a fixed discriminator, Labeler and Anti-Labeler, the generator solves the following optimization problem:

$$\begin{aligned}
&\min_G \mathbb{E}_{x\sim p_{\text{data}}(x)} \left[\log(D(x))\right] + \mathbb{E}_{x\sim p_{\text{g}}(x)} \left[\log\left(\frac{1-D(x)}{D(x)}\right)\right] \\
&- \frac{1}{d}\sum_{j=1}^{d} \rho_j \mathbb{E}_{x\sim\mathbb{P}_g(x|l_j=1)} \left[\log(D_{LR}(X)[j])\right] \\
&- (1-\rho_j)\mathbb{E}_{x\sim\mathbb{P}_g(x|l_j=0)} \left[\log(1-D_{LR}(X)[j])\right] \\
&+ \frac{1}{d}\sum_{j=1}^{d} \rho_j \mathbb{E}_{x\sim\mathbb{P}_g(x|l_j=1)} \left[\log(D_{LG}(X)[j])\right] \\
&+ (1-\rho_j)\mathbb{E}_{x\sim\mathbb{P}_g(x|l_j=0)} \left[\log(1-D_{LG}(X)[j])\right].
\end{aligned} \tag{E.16}$$

We have the following proposition, which characterizes the optimum generator for optimum Labeler, Anti-Labeler and Discriminator:

Proposition 8. *Define $C(G)$ as the generator loss for when discriminator, Labeler and Anti-Labeler are at their optimum obtained from (E.16). The global minimum of the virtual training criterion $C(G)$ is achieved if and only if $\mathbb{P}_g(x|l_i) = \mathbb{P}_r(x|l_i) \forall i \in [d]$ and $\mathbb{P}_g(x) = \mathbb{P}_r(x)$.*

Proof. Proof follows the same lines as in the proof of Theorem 17 and Theorem 19 and is omitted. □

Thus we have

$$\mathbb{P}_r(x, l_i) = \mathbb{P}_g(x, l_i), \forall i \in [d] \text{ and } \mathbb{P}_r(x) = \mathbb{P}_g(x). \tag{E.17}$$

However, this does not in general imply $\mathbb{P}_r(x, l_1, l_2, \ldots, l_d) = \mathbb{P}_g(x, l_1, l_2, \ldots, l_d)$, which is equivalent to saying the generated distribution samples from the class conditional image distributions. To guarantee the correct conditional sampling given all labels, we introduce the following assumption: We assume that the image x determines all the labels. This assumption is very relevant in practice. For example, in the CelebA dataset, which we use, the label vector, e.g., whether the person is a male or female, with or without a mustache, can be thought of as a deterministic function of the image. When this is true, we can say that $\mathbb{P}_r(l_1, l_2, \ldots, l_n|x) = \mathbb{P}_r(l_1|x)\mathbb{P}_r(l_2|x) \ldots \mathbb{P}_r(l_n|x)$.

We need the following lemma, where kronecker delta function refers to the functions that take the value of 1 only on a single point, and 0 everywhere

else:

Lemma 26. *Any discrete joint probability distribution, where all the marginal probability distributions are kronecker delta functions is the product of these marginals.*

Proof. Let $\delta_{\{x-u\}}$ be the kronecker delta function which is 1 if $x = u$ and is 0 otherwise. Consider a joint distribution $p(X_1, X_2, \ldots, X_n)$, where $p(X_i) = \delta_{\{X_i - u_i\}}, \forall i \in [n]$, for some set of elements $\{u_i\}_{i\in[n]}$. We will show by contradiction that the joint probability distribution is zero everywhere except at $(u_1, u_2, \ldots, u_n)$. Then, for the sake of contradiction, suppose for some $v = (v_1, v_2, \ldots, v_n) \neq (u_1, u_2, \ldots, u_n)$, $p(v_1, v_2, \ldots, v_n) \neq 0$. Then $\exists j \in [n]$ such that $v_j \neq u_j$. Then we can marginalize the joint distribution as

$$p(v_j) = \sum_{X_1,\ldots,X_{j-1},X_j,\ldots,X_n} p(X_1, \ldots, X_{j-1}, v_j, X_{j+1}, \ldots, X_n) > 0, \qquad \text{(E.18)}$$

where the inequality is due to the fact that the particular configuration $(v_1, v_2, \ldots, v_n)$ must have contributed to the summation. However this contradicts with the fact that $p(X_j) = 0, \forall X_j \neq u_j$. Hence, $p(.)$ is zero everywhere except at $(u_1, u_2, \ldots, u_n)$, where it should be 1. □

We can now simply apply the above lemma on the conditional distribution $\mathbb{P}_g(l_1, l_2, \ldots, l_d|x)$. Proposition 8 shows that the image distributions and the marginals $\mathbb{P}_g(l_i|x)$ are true to the data distribution due to Bayes' rule. Since the vector $(l_1, \ldots, l_n)$ is a deterministic function of x by assumption, $\mathbb{P}_r(l_i|x)$ are kronecker delta functions, and so are $\mathbb{P}_g(l_i|x)$ by Proposition 8. Thus,

since the joint $\mathbb{P}_g(x, l_1, l_2, \ldots, l_d)$ satisfies the condition that every marginal distribution $p(l_i|x)$ is a kronecker delta function, then it must be a product distribution by Lemma 26. Thus we can write

$$\mathbb{P}_g(l_1, l_2, \ldots, l_d|x) = \mathbb{P}_g(l_1|x)\mathbb{P}_g(l_2|x)\ldots\mathbb{P}_g(l_n|x).$$

Then we have the following chain of equalities.

$$\begin{aligned}
\mathbb{P}_r(x, l_1, l_2, \ldots, l_d) &= \mathbb{P}_r(l_1, \ldots, l_n|x)\mathbb{P}_r(x) \\
&= \mathbb{P}_r(l_1|x)\mathbb{P}_r(l_2|x)\ldots\mathbb{P}_r(l_n|x)\mathbb{P}_r(x) \\
&= \mathbb{P}_g(l_1|x)\mathbb{P}_g(l_2|x)\ldots\mathbb{P}_g(l_n|x)\mathbb{P}_g(x) \\
&= \mathbb{P}_g(l_1, l_2, \ldots, l_d|x)\mathbb{P}_g(x) \\
&= \mathbb{P}_g(x, l_1, l_2, \ldots, l_d).
\end{aligned}$$

Thus, we also have $\mathbb{P}_r(x|l_1, l_2, \ldots, l_n) = \mathbb{P}_g(x|l_1, l_2, \ldots, l_n)$ since

$$\mathbb{P}_r(l_1, l_2, \ldots, l_n) = \mathbb{P}_g(l_1, l_2, \ldots, l_n),$$

concluding the proof that the optimum generator samples from the class conditional image distributions.

E.8 CausalBEGAN Architecture

In this section, we propose a simple, but non-trivial extension of BEGAN where we feed image labels to the generator. One of the central contributions of BEGAN ([13]) is a control theory-inspired boundary equilibrium approach that encourages generator training only when the discriminator is near optimum and

its gradients are the most informative. The following observation helps us carry the same idea to the case with labels: Label gradients are most informative when the image quality is high. Here, we introduce a new loss and a set of margins that reflect this intuition.

Formally, let $\mathcal{L}(x)$ be the average L_1 pixel-wise autoencoder loss for an image x, as in BEGAN. Let $\mathcal{L}_{sq}(u, v)$ be the squared loss term, i.e., $\|u - v\|_2^2$. Let (x, l_x) be a sample from the data distribution, where x is the image and l_x is its corresponding label. Similarly, $G(z, l_g)$ is an image sample from the generator, where l_g is the label used to generate this image. Denoting the space of images by $\mathcal{I}$, let $G : \mathbb{R}^n \times \{0, 1\}^m \mapsto \mathcal{I}$ be the generator. As a naive attempt to extend the original BEGAN loss formulation to include the labels, we can write the following loss functions:

$$\begin{aligned} Loss_D &= \mathcal{L}(x) - \mathcal{L}(Labeler(G(z, l))) + \mathcal{L}_{sq}(l_x, Labeler(x)) \\ &\qquad - \mathcal{L}_{sq}(l_g, Labeler(G(z, l_g))), \\ Loss_G &= \mathcal{L}(G(z, l_g)) + \mathcal{L}_{sq}(l_g, Labeler(G(z, l_g))). \end{aligned} \tag{E.19}$$

However, this naive formulation does not address the use of margins, which is extremely critical in the BEGAN formulation. Just as a better trained BEGAN discriminator creates more useful gradients for image generation, a better trained Labeler is a prerequisite for meaningful gradients. This motivates an additional margin-coefficient tuple (b_2, c_2), as shown in (E.20,E.21).

The generator tries to jointly minimize the two loss terms in the formulation in (E.19). We empirically observe that occasionally the image quality

will suffer because the images that best exploit the Labeler network are often not obliged to be realistic, and can be noisy or misshapen. Based on this, label loss seems unlikely to provide useful gradients unless the image quality remains good. Therefore we encourage the generator to incorporate label loss only when the *image quality margin* b_1 is large compared to the *label margin* b_2. To achieve this, we introduce a new *margin of margins* term, b_3. As a result, the margin equations and update rules are summarized as follows, where $\lambda_1, \lambda_2, \lambda_3$ are learning rates for the coefficients.

$$\begin{aligned} b_1 &= \gamma_1 * \mathcal{L}(x) - \mathcal{L}(G(z, l_g)). \\ b_2 &= \gamma_2 * \mathcal{L}_{sq}(l_x, Labeler(x)) - \mathcal{L}_{sq}(l_g, Labeler(G(z, l_g))). \\ b_3 &= \gamma_3 * relu(b_1) - relu(b_2). \end{aligned} \tag{E.20}$$

$$\begin{aligned} c_1 &\leftarrow clip_{[0,1]}(c_1 + \lambda_1 * b_1). \\ c_2 &\leftarrow clip_{[0,1]}(c_2 + \lambda_2 * b_2). \\ c_3 &\leftarrow clip_{[0,1]}(c_3 + \lambda_3 * b_3). \end{aligned} \tag{E.21}$$

$$Loss_D = \mathcal{L}(x) - c_1 * \mathcal{L}(G(z, l_g)) + \mathcal{L}_{sq}(l_x, Labeler(x)) - c_2 * \mathcal{L}_{sq}(l_g, G(z, l_g)). \tag{E.22}$$

$$Loss_G = \mathcal{L}(G(z, l_g)) + c_3 * \mathcal{L}_{sq}(l_g, Labeler(G(z, l_g))).$$

One of the advantages of BEGAN is the existence of a monotonically decreasing scalar which can track the convergence of the gradient descent optimization. Our extension preserves this property as we can define

$$\mathcal{M}_{complete} = \mathcal{L}(x) + |b_1| + |b_2| + |b_3|, \tag{E.23}$$

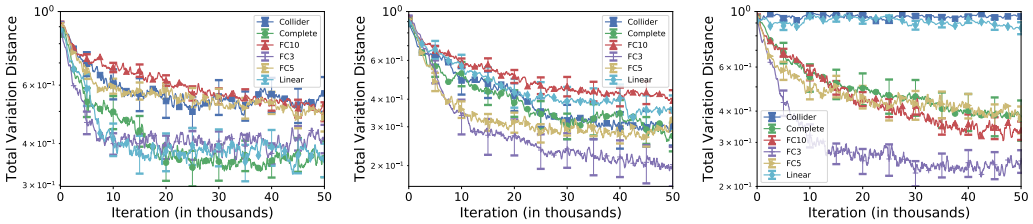
and show that $\mathcal{M}_{complete}$ decreases progressively during our optimizations. See Figure E.12.

E.9 Dependence of GAN Behavior on Causal Graph

In Section 6.4 we showed how a GAN could be used to train a causal implicit generative model by incorporating the causal graph into the generator structure. Here we investigate the behavior and convergence of causal implicit generative models when the true data distribution arises from another (possibly distinct) causal graph.

We consider causal implicit generative model convergence on synthetic data whose three features $\{X, Y, Z\}$ arise from one of three causal graphs: "line" $X \to Y \to Z$, "collider" $X \to Y \leftarrow Z$, and "complete" $X \to Y \to Z, X \to Z$. For each node a (randomly sampled once) cubic polynomial in $n + 1$ variables computes the value of that node given its n parents and 1 uniform exogenous variable. We then repeat, creating a new synthetic dataset in this way for each causal model and report the averaged results of 20 runs for each model.

For each of these data generating graphs, we compare the convergence of the joint distribution to the true joint in terms of the total variation distance, when the generator is structured according to a line, collider, or complete graph. For completeness, we also include generators with no knowledge of causal structure: $\{fc3, fc5, fc10\}$ are fully connected neural networks that map uniform random noise to 3 output variables using either 3,5, or 10 layers respectively.



(a) $X \to Y \to Z$ (b) $X \to Y \leftarrow Z$ (c) $X \to Y \to Z$, $X \to Z$

Figure E.2: Convergence in total variation distance of generated distribution to the true distribution for causal implicit generative model, when the generator is structured based on different causal graphs. (a) Data generated from line graph $X \to Y \to Z$. The best convergence behavior is observed when the true causal graph is used in the generator architecture. (b) Data generated from collider graph $X \to Y \leftarrow Z$. Fully connected layers may perform better than the true graph depending on the number of layers. Collider and complete graphs performs better than the line graph which implies the wrong Bayesian network. (c) Data generated from complete graph $X \to Y \to Z$, $X \to Z$. Fully connected with 3 layers performs the best, followed by the complete and fully connected with 5 and 10 layers. Line and collider graphs, which implies the wrong Bayesian network does not show convergence behavior.

The results are given in Figure E.2. Data is generated from line causal graph $X \to Y \to Z$ (left panel), collider causal graph $X \to Y \leftarrow$ (middle panel), and complete causal graph $X \to Y \to Z, X \to Z$ (right panel). Each curve shows the convergence behavior of the generator distribution, when generator is structured based on each one of these causal graphs. We expect convergence when the causal graph used to structure the generator is capable of generating the joint distribution due to the true causal graph: as long as we use the correct Bayesian network, we should be able to fit to the true joint. For example, complete graph can encode all joint distributions. Hence, we expect complete graph to work well with all data generation models. Standard fully connected layers correspond to the causal graph with a latent variable causing all the observable variables. Ideally, this model should be able to fit to any causal generative model. However, the convergence behavior of adversarial training across these models is unclear, which is what we are exploring with Figure E.2.

For the line graph data $X \to Y \to Z$, we see that the best convergence behavior is when line graph is used in the generator architecture. As expected, complete graph also converges well, with slight delay. Similarly, fully connected network with 3 layers show good performance, although surprisingly fully connected with 5 and 10 layers perform much worse. It seems that although fully connected can encode the joint distribution in theory, in practice with adversarial training, the number of layers should be tuned to achieve the same performance as using the true causal graph. Using the wrong Bayesian network,

the collider, also yields worse performance.

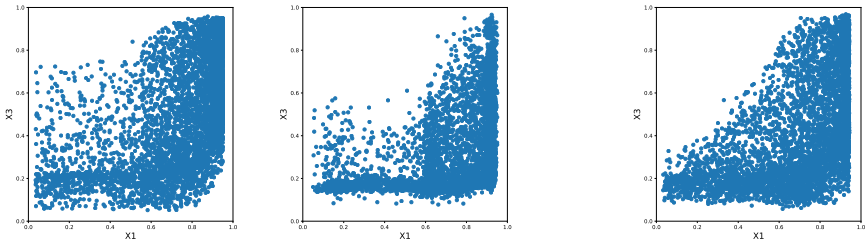
For the collider graph, surprisingly using a fully connected generator with 3 and 5 layers shows the best performance. However, consistent with the previous observation, the number of layers is important, and using 10 layers gives the worst convergence behavior. Using complete and collider graphs achieves the same decent performance, whereas line graph, a wrong Bayesian network, performs worse than the two.

For the complete graph, fully connected 3 performs the best, followed by fully connected 5, 10 and the complete graph. As we expect, line and collider graphs, which cannot encode all the distributions due to a complete graph, performs the worst and does not actually show any convergence behavior.

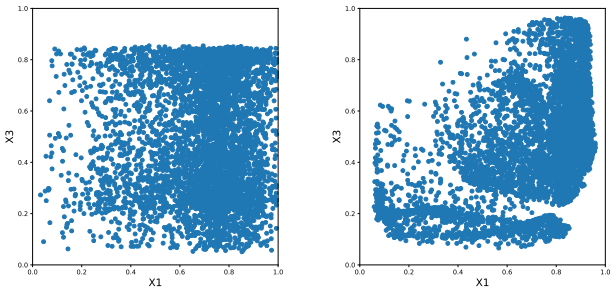
E.10 Additional Simulations for Causal Controller

First, we evaluate the effect of using the wrong causal graph on an artificially generated dataset. Figure E.3 shows the scatter plot for the two coordinates of a three dimensional distribution. As we observe, using the correct graph gives the closest scatter plot to the original data, whereas using the wrong Bayesian network, collider graph, results in a very different distribution.

Second, we expand on the causal graphs used for experiments for the CelebA dataset. We use a causal graph on a subset of the image labels of CelebA dataset, which we call CelebA Causal Graph (G1), illustrated in Figure E.1. The graph cG1, which is a completed version of G1, is the complete



(a) $X_1 \to X_2 \to X_3$ (b) $X_1 \to X_2 \to X_3$ (c) $X_1 \to X_2 \to X_3$, $X_1 \to X_3$



(d) $X_1 \to X_2 \leftarrow X_3$ (e) Fully connected

Figure E.3: Synthetic data experiments: (a) Scatter plot for actual data. Data is generated using the causal graph $X_1 \to X_2 \to X_3$. (b) Generated distribution when generator causal graph is $X_1 \to X_2 \to X_3$. (c) Generated distribution when generator causal graph is $X_1 \to X_2 \to X_3 \cup X_1 \to X_3$. (d) Generated distribution when generator causal graph is $X_1 \to X_2 \leftarrow X_3$. (e) Generated distribution when generator is from a fully connected last layer of a 5 layer FF neural net.

Label Pair		Male	
		0	1
Young	0	0.14[0.07](0.07)	0.09[0.15](0.15)
	1	0.47[0.51](0.51)	0.29[0.27](0.26)
Mustache	0	0.61[0.58](0.58)	0.34[0.38](0.38)
	1	0.00[0.00](0.00)	0.04[0.04](0.04)

Table E.1: Pairwise marginal distribution for select label pairs when Causal Controller is trained on $G1$ in plain text, its completion $cG1$[square brackets], and the true pairwise distribution(in parentheses). Note that $G1$ treats Male and Young labels as independent, but does not completely fail to generate a reasonable (product of marginals) approximation. Also note that when an edge is added $Young \rightarrow Male$, the learned distribution is nearly exact. Note that both graphs contain the edge $Male \rightarrow Mustache$ and so are able to learn that women have no mustaches.

graph associated with the ordering: Young, Male, Eyeglasses, Bald, Mustache, Smiling, Wearing Lipstick, Mouth Slightly Open, Narrow Eyes. For example, in cG1 Male causes Smiling because Male comes before Smiling in the ordering. The graph rcG1 is formed by reversing every edge in cG1.

Next, we check the effect of using the incorrect Bayesian network for the data. The causal graph G1 generates Male and Young independently, which is incorrect in the data. Comparison of pairwise distributions in Table E.1 demonstrate that for G1 a reasonable approximation to the true distribution is still learned for {Male, Young} jointly. For cG1 a nearly perfect distributional approximation is learned. Furthermore we show that despite this inaccuracy, both graphs G1 and cG1 lead to Causal Controllers that never output the label combination {Female,Mustache}, which will be important later.

Wasserstein GAN in its original form (with Lipshitz discriminator)

assures convergence in distribution of the Causal Controller output to the discretely supported distribution of labels. We use a slightly modified version of Wasserstein GAN with a penalized gradient ([48]). We first demonstrate that learned outputs actually have "approximately discrete" support. In Figure E.4a, we sample the joint label distribution 1000 times, and make a histogram of the (all) scalar outputs corresponding to any label.

Although Figure E.4b demonstrates conclusively good convergence for both graphs, TVD is not always intuitive. For example, "how much can each marginal be off if there are 9 labels and the TVD is 0.14?". To expand upon Figure E.2 where we showed that the causal controller learns the correct distribution for a pairwise subset of nodes, here we also show that both CelebA Causal Graph (G1) and the completion we define (cG1) allow training of very reasonable marginal distributions for all labels (Table E.1) that are not off by more than 0.03 for the worst label. $\mathbb{P}_D(L = 1)$ is the probability that the label is 1 in the dataset, and $\mathbb{P}_G(L = 1)$ is the probability that the generated label is (around a small neighborhood of) 1.

E.11 Wasserstein Causal Controller on CelebA Labels

We test the performance of our Wasserstein Causal Controller on a subset of the binary labels of CelebA datset. We use the causal graph given in Figure E.1.

For causal graph training, first we verify that our Wasserstein training allows the generator to learn a mapping from continuous uniform noise to a

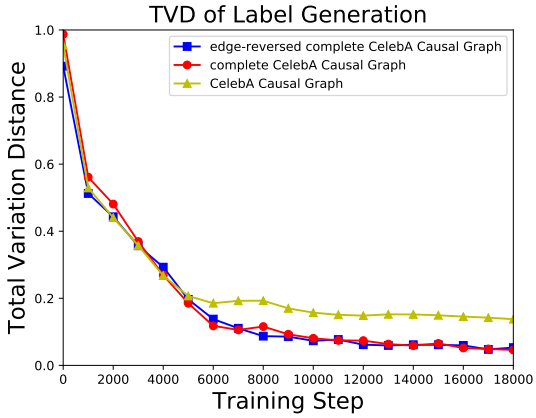
Label, L	$\mathbb{P}_{G1}(L=1)$	$\mathbb{P}_{cG1}(L=1)$	$\mathbb{P}_{D}(L=1)$
Bald	0.02244	0.02328	0.02244
Eyeglasses	0.06180	0.05801	0.06406
Male	0.38446	0.41938	0.41675
Mouth Slightly Open	0.49476	0.49413	0.48343
Mustache	0.04596	0.04231	0.04154
Narrow Eyes	0.12329	0.11458	0.11515
Smiling	0.48766	0.48730	0.48208
Wearing Lipstick	0.48111	0.46789	0.47243
Young	0.76737	0.77663	0.77362

Table E.2: Marginal distribution of pretrained Causal Controller labels when Causal Controller is trained on CelebA Causal Graph (P_{G1}) and its completion(P_{cG1}), where $cG1$ is the (nonunique) largest DAG containing $G1$ (see appendix). The third column lists the actual marginal distributions in the dataset

discrete distribution. Figure E.4a shows where the samples, averaged over all the labels in CelebA Causal Graph, from this generator appears on the real line. The result emphasizes that the proposed Causal Controller outputs an almost discrete distribution: 96% of the samples appear in $0.05-$neighborhood of 0 or 1. Outputs shown are *unrounded* generator outputs.

A stronger measure of convergence is the total variational distance (TVD). For CelebA Causal Graph (G1), our defined completion (cG1), and cG1 with arrows reversed (rcG1), we show convergence of TVD with training (Figure E.4b). Both cG1 and rcG1 have TVD decreasing to 0, and TVD for G1 assymptotes to around 0.14 which corresponds to the incorrect conditional independence assumptions that G1 makes. This suggests that any given complete causal graph will lead to a nearly perfect implicit causal generator

(a) Essentially Discrete Range of Causal Controller



(b) TVD vs. No. of Iters in CelebA Labels

Figure E.4: (a) A number line of unit length binned into 4 unequal bins along with the percent of Causal Controller ($G1$) samples in each bin. Results are obtained by sampling the joint label distribution 1000 times and forming a histogram of the scalar outputs corresponding to any label. Note that our Causal Controller output labels are approximately discrete even though the input is a continuum (uniform). The 4% between 0.05 and 0.95 is not at all uniform and almost zero near 0.5. (b) Progression of total variation distance between the Causal Controller output with respect to the number of iterations: CelebA Causal Graph is used in the training with Wasserstein loss.

Intervening vs Conditioning on Wearing Lipstick, Top: Intervene Wearing Lipstick=1, Bottom: Condition Wearing Lipstick=1

Figure E.5: Intervening/Conditioning on Wearing Lipstick label in CelebA Causal Graph. Since $Male \rightarrow$ *Wearing Lipstick* in CelebA Causal Graph, we do not expect $do($*Wearing Lipstick*$= 1)$ to affect the probability of $Male = 1$, i.e., $\mathbb{P}(Male = 1|do($*Wearing Lipstick*$= 1)) = \mathbb{P}(Male = 1) = 0.42$. Accordingly, the top row shows both males and females who are wearing lipstick. However, the bottom row of images sampled from the conditional distribution $\mathbb{P}(.|$*Wearing Lipstick* $= 1)$ shows only female images because in the dataset $\mathbb{P}(Male = 0|$*Wearing Lipstick* $= 1) \approx 1$.

over labels and that bayesian partially incorrect causal graphs can still give reasonable convergence.

E.12 More CausalGAN Results

In this section, we present additional CausalGAN results in Figure E.5, E.6.

E.13 More CausalBEGAN Results

In this section, we train CausalBEGAN on CelebA dataset using CelebA Causal Graph. The Causal Controller is pretrained with a Wasserstein loss and used for training the CausalBEGAN.

To first empirically justify the need for the margin of margins we

Intervening vs Conditioning on Narrow Eyes, Top: Intervene Narrow Eyes=1, Bottom: Condition Narrow Eyes=1

Figure E.6: Intervening/Conditioning on Narrow Eyes label in CelebA Causal Graph. Since $Smiling \to Narrow\ Eyes$ in CelebA Causal Graph, we do not expect $do(Narrow\ Eyes = 1)$ to affect the probability of $Smiling = 1$, i.e., $\mathbb{P}(Smiling = 1|do(Narrow\ Eyes = 1)) = \mathbb{P}(Smiling = 1) = 0.48$. However on the bottom row, conditioning on $Narrow\ Eyes = 1$ increases the proportion of smiling images (From 0.48 to 0.59 in the dataset), although 10 images may not be enough to show this difference statistically.

introduced in (E.22) (c_3 and b_3), we train the same CausalBEGAN model setting $c_3 = 1$, removing the effect of this margin. We show that the image quality for rare labels deteriorates. Please see Figure E.11 in the appendix. Then for the labels *Bald*, and *Mouth Slightly Open*, we illustrate the difference between interventional and conditional sampling when the label is 1. (Figures E.7, E.8).

E.14 Label Sweeping and Diversity for CausalGAN

In this section, we provide additional simulations for CausalGAN. In Figures E.9a-E.9d, we show the conditional image generation properties of CausalGAN by sweeping a single label from 0 to 1 while keeping all other inputs/labels fixed. In Figure E.10, to examine the degree of mode collapse and show the image diversity, we show 256 randomly sampled images.

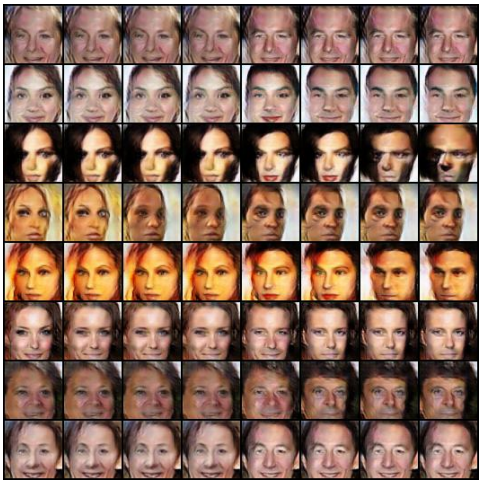
Intervening vs Conditioning on Bald, Top: Intervene Bald=1, Bottom: Condition Bald=1

Figure E.7: Intervening/Conditioning on Bald label in CelebA Causal Graph. Since $Male \rightarrow Bald$ in CelebA Causal Graph, we do not expect $do(Bald = 1)$ to affect the probability of $Male = 1$, i.e., $\mathbb{P}(Male = 1|do(Bald = 1)) = \mathbb{P}(Male = 1) = 0.42$. Accordingly, the top row shows both bald males and bald females. The bottom row of images sampled from the conditional distribution $\mathbb{P}(.|Bald = 1)$ shows only male images because in the dataset $\mathbb{P}(Male = 1|Bald = 1) \approx 1$.

Intervening vs Conditioning on Mouth Slightly Open, Top: Intervene Mouth Slightly Open=1, Bottom: Condition Mouth Slightly Open=1

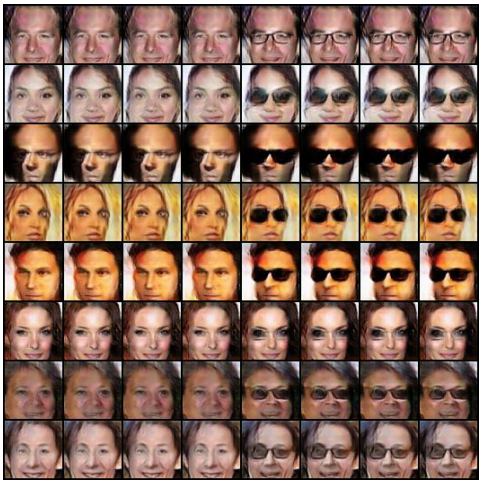
Figure E.8: Intervening/Conditioning on Mouth Slightly Open label in CelebA Causal Graph. Since $Smiling \rightarrow MouthSlightlyOpen$ in CelebA Causal Graph, we do not expect $do(Mouth\ Slightly\ Open = 1)$ to affect the probability of $Smiling = 1$, i.e., $\mathbb{P}(Smiling = 1|do(Mouth\ Slightly\ Open = 1)) = \mathbb{P}(Smiling = 1) = 0.48$. However on the bottom row, conditioning on $Mouth\ Slightly\ Open = 1$ increases the proportion of smiling images (From 0.48 to 0.76 in the dataset), although 10 images may not be enough to show this difference statistically.

Interpolating Bald label



Interpolating Male label

Interpolating Young label



Interpolating Eyeglasses label

Figure E.9: The effect of interpolating a single label for CausalGAN, while keeping the noise terms and other labels fixed.

Figure E.10: Diversity of the proposed CausalGAN showcased with 256 samples.

E.15 Additional CausalBEGAN Simulations

In this section, we provide additional simulation results for CausalBEGAN. First we show that although our third margin term b_3 introduces complications, it can not be ignored. Figure E.11 demonstrates that omitting the third margin on the image quality of rare labels.

Furthermore just as the setup in BEGAN permitted the definiton of a scalar "$\mathcal{M}$", which was monotonically decreasing during training, our definition permits an obvious extension $\mathcal{M}_{complete}$ (defined in E.23) that preserves these properties. See Figure E.12 to observe $\mathcal{M}_{complete}$ decreaing monotonically during training.

We also show the conditional image generation properties of CausalBEGAN by using "label sweeps" that move a single label input from 0 to 1 while keeping all other inputs fixed (Figures E.13a -E.13d). It is interesting to note that while generators are often implicitly thought of as continuous functions, the generator in this CausalBEGAN architecture learns a discrete function with respect to its label input parameters. (Initially there is label interpolation, and later in the optimization label interpolation becomes more step function like (not shown)). Finally, to examine the degree of mode collapse and show the image diversity, we show a random sampling of 256 images (Figure E.14).

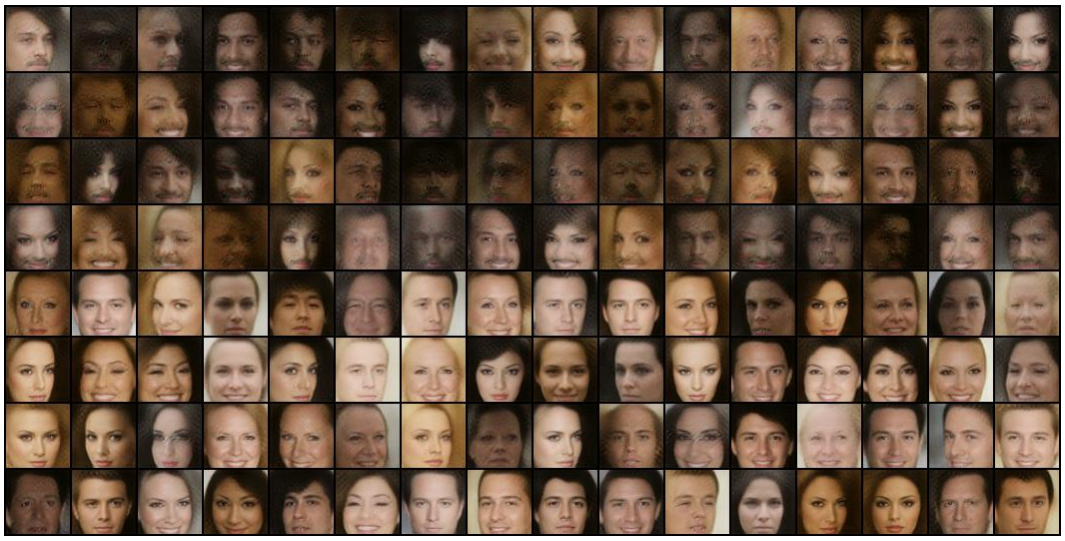


Figure E.11: Omitting the nonobvious margin $b_3 = \gamma_3 * relu(b_1) - relu(b_2)$ results in poorer image quality particularly for rare labels such as mustache. We compare samples from two interventional distributions. Samples from $\mathbb{P}(.|do(Mustache = 1))$ (top) have much poorer image quality compared to those under $\mathbb{P}(.|do(Mustache = 0))$ (bottom).

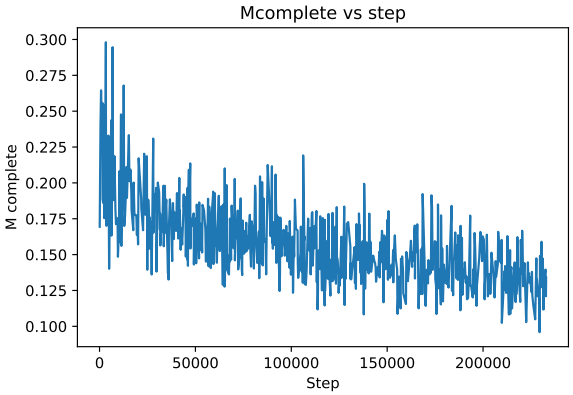
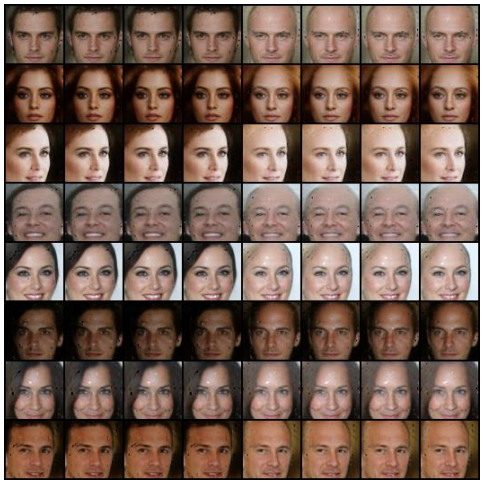
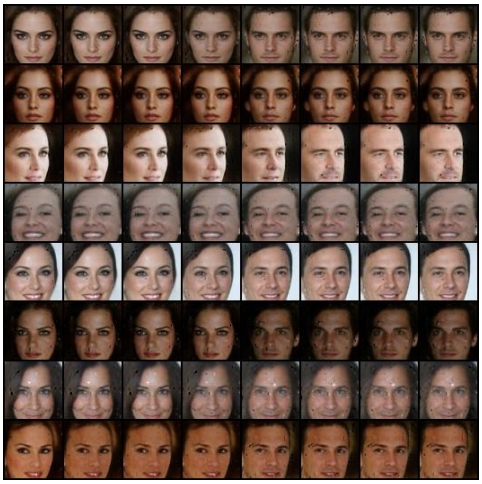


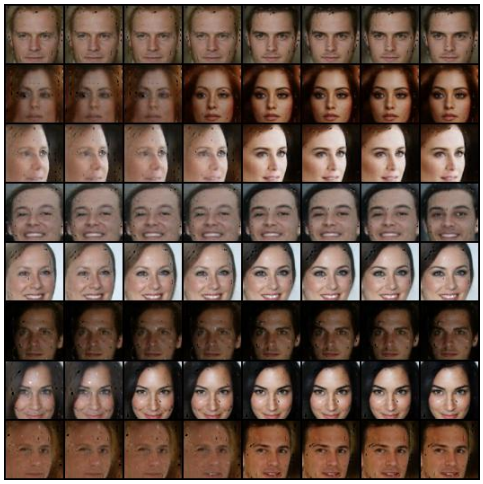
Figure E.12: Convergence of CausalBEGAN captured through the parameter $\mathcal{M}_{complete}$.



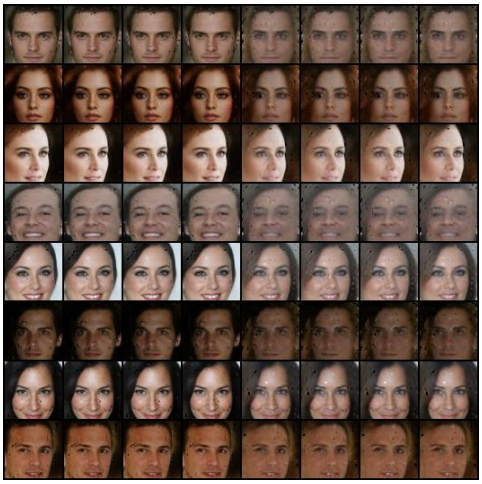
Interpolating Bald label



Interpolating Male label



Interpolating Young label



Interpolating Eyeglasses label

Figure E.13: The effect of interpolating a single label for CausalBEGAN, while keeping the noise terms and other labels fixed. Although most labels are properly captured, we see that eyeglasses label is not.

Figure E.14: Diversity of Causal BEGAN showcased with 256 samples.

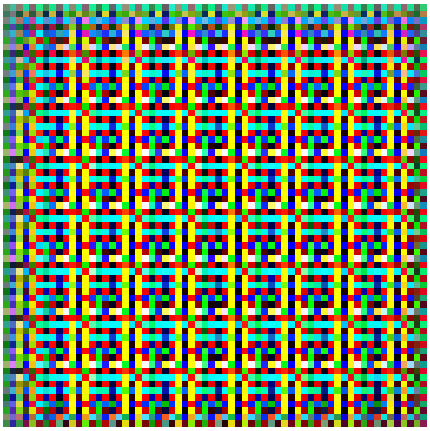


Figure E.15: Failed Image generation for simultaneous label and image generation after 20k steps.

E.16 Directly Training CiGM for Labels+Image Fails

In this section, we present the result of attempting to jointly train an implicit causal generative model for labels and the image. This approach treats the image as part of the causal graph. It is not clear how exactly to feed both labels and image to discriminator, but one way is to simply encode the label as a constant image in an additional channel. We tried this for CelebA Causal Graph and observed that the image generation is not learned (Figure E.15). One hypothesis is that the discriminator focuses on labels without providing useful gradients to the image generation.

E.17 Implementation

In this section, we explain the differences between implementation and theory, along with other implementation details for both CausalGAN and

CausalBEGAN.

E.18 Pretraining Causal Controller for Face Labels

In this section, we explain the implementation details of the Wasserstein Causal Controller for generating face labels. We used the total variation distance (TVD) between the distribution of generator and data distribution as a metric to decide the success of the models.

The gradient term used as a penalty is estimated by evaluating the gradient at points interpolated between the real and fake batches. Interestingly, this Wasserstein approach gives us the opportunity to train the Causal Controller to output (almost) discrete labels (See Figure E.4a). In practice though, we still found benefit in rounding them before passing them to the generator.

The generator architecture is structured in accordance with Section 6.4 based on the causal graph in Figure E.1, using uniform noise as exogenous variables and 6 layer neural networks as functions mapping parents to children. For the training, we used 25 Wasserstein discriminator (critic) updates per generator update, with a learning rate of 0.0008.

E.19 Implementation Details for CausalGAN

In practice, we use stochastic gradient descent to train our model. We use *DCGAN* [111], a convolutional neural net-based implementation of generative adversarial networks, and extend it into our Causal GAN framework.

We have expanded it by adding our Labeler networks, training a Causal Controller network and modifying the loss functions appropriately. Compared to DCGAN an important distinction is that we make 6 generator updates for each discriminator update on average. The discriminator and labeler networks are concurrently updated in a single iteration.

Notice that the loss terms defined in Section 6.5.2.1 contain a single binary label. In practice we feed a d-dimensional label vector and need a corresponding loss function. We extend the Labeler and Anti-Labeler loss terms by simply averaging the loss terms for every label. The i^{th} coordinates of the d-dimensional vectors given by the labelers determine the loss terms for label i. Note that this is different than the architecture given in Section E.6, where the discriminator outputs a length-2^d vector and estimates the probabilities of all label combinations given the image. Therefore this approach does not have the guarantee to sample from the class conditional distributions, if the data distribution is not restricted. However, for the type of labeled image dataset we use in this work, where labels seem to be completely determined given an image, this architecture is sufficient to have the same guarantees. For the details, please see Section E.7 in the supplementary material.

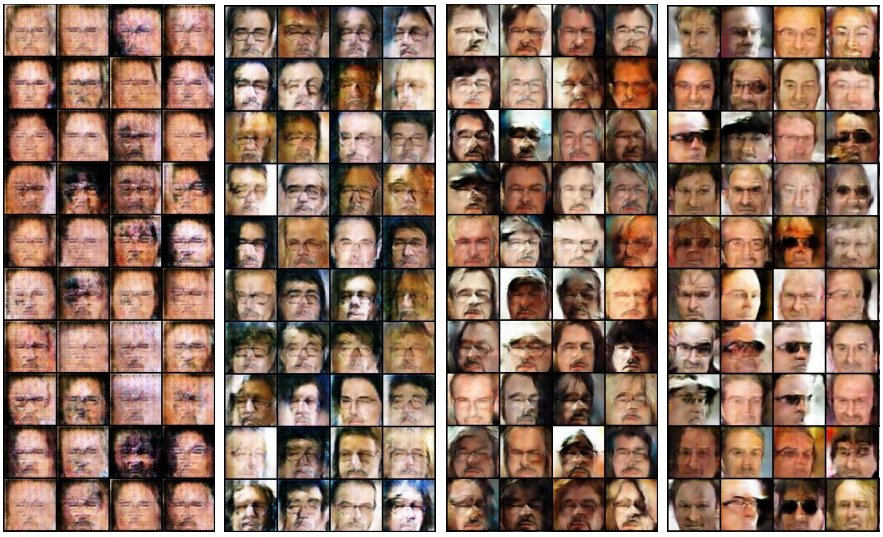
Compared to the theory we have, another difference in the implementation is that we have swapped the order of the terms in the cross entropy expressions for labeler losses. This has provided sharper images at the end of the training.

E.20 Conditional Image Generation for CausalBEGAN

The labels input to CausalBEGAN are taken from the Causal Controller. We use very few parameter tunings. We use the same learning rate (0.00008) for both the generator and discriminator and do 1 update of each simultaneously (calculating the for each before applying either). We simply use $\gamma_1 = \gamma_2 = \gamma_3 = 0.5$. We do not expect the model to be very sensitive to these parameter values, as we achieve good performance without hyperparameter tweaking. We do use customized margin learning rates $\lambda_1 = 0.001, \lambda_2 = 0.00008, \lambda_3 = 0.01$, which reflect the asymmetry in how quickly the generator can respond to each margin. For example c_2 can have much more "spiky", fast responding behavior compared to others even when paired with a smaller learning rate, although we have not explored this parameter space in depth. In these margin behaviors, we observe that the best performing models have all three margins "active": near 0 while frequently taking small positive values.

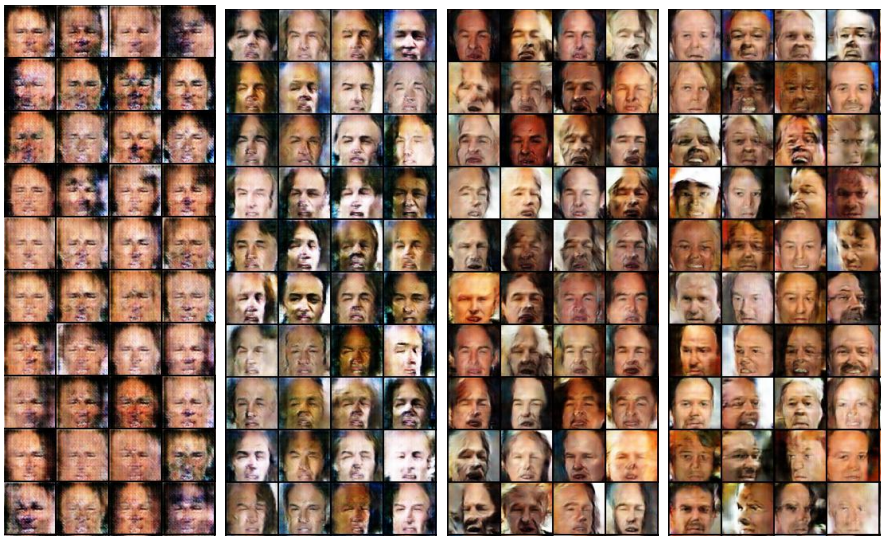
E.21 Role of Anti-Labeler

In this section, we show results that compare the CausalGAN behavior with and without Anti-Labeler network. In general, using Anti-Labeler allows for faster convergence. For very rare labels, the model with Anti-Labeler provides more diverse images. See Figures E.16, E.17, E.18.



(a) No AL, $t = 20k$ (b) No AL, $t = 30k$ (c) No AL, $t = 40k$ (d) No AL, $t = 20k$

Figure E.16: CausalGAN results with and without Anti-Labeler for the rare label combination *Old males with eyeglasses and mustache and narrow eyes who are not smiling*. (a, b, c) Samples without Anti-Labeler at iterations $20k, 30k, 40k$ respectively. (d) Samples with Anti-Labeler at iteration $20k$. Comparing (a) and (d), we observe that using Anti-Labeler allows for faster convergence. Comparing (c) and (d), we observe that using Anti-Labeler provides more diverse images.



(a) No AL, $t = 20k$ (b) No AL, $t = 30k$ (c) No AL, $t = 40k$ (d) No AL, $t = 20k$

Figure E.17: CausalGAN results with and without Anti-Labeler for the rare label combination *Old bald males who are not smiling but have an open mouth and narrow eyes*. (a, b, c) Samples without Anti-Labeler at iterations $20k, 30k, 40k$ respectively. (d) Samples with Anti-Labeler at iteration $20k$. Comparing (a) and (d), we observe that using Anti-Labeler allows for faster convergence.

(a) No AL, $t = 20k$ (b) No AL, $t = 30k$ (c) No AL, $t = 40k$ (d) No AL, $t = 20k$

Figure E.18: CausalGAN results with and without Anti-Labeler for the common label combination *Young smiling women with lipstick*. (a, b, c) Samples without Anti-Labeler at iterations $20k, 30k, 40k$ respectively. (d) Samples with Anti-Labeler at iteration $20k$. Comparing (a) and (d), we observe that using Anti-Labeler allows for faster convergence.

Bibliography

[1] Alfred V. Aho, Michael R Garey, and Jeffrey D. Ullman. The transitive reduction of a directed graph. *SIAM Journal on Computing*, 1(2):131–137, 1972.

[2] Ayesha R. Ali, Thomas S. Richardson, Peter L. Spirtes, and Jiji Zhang. Towards characterizing markov equivalence classes for directed acyclic graphs with latent variables. In *Proc. of the Uncertainty in Artificial Intelligence*, 2005.

[3] Noga Alon. Covering graphs by the minimum number of equivalence relations. *Combinatorica*, 6(3):201–206, 1986.

[4] Steen A. Andersson, David Madigan, and Michael D. Perlman. A characterization of markov equivalence classes for acyclic digraphs. *The Annals of Statistics*, 25(2):505–541, 1997.

[5] Grigory Antipov, Moez Baccouche, and Jean-Luc Dugelay. Face aging with conditional generative adversarial networks. In *IEEE International Conference on Image Processing (ICIP)*, Sept. 2017.

[6] Martin Arjovsky and Léon Bottou. Toward principled methods for training generative adversarial networks. In *Proc. of ICLR'17*, 2017.

[7] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan. In *Proc. NIPS 2017*, 2017.

[8] Sanjeev Arora, Rong Ge, Yonatan Halpern, David Mimno, Ankur Moitra, David Sontag, Yichen Wu, and Michael Zhu. A practical algorithm for topic modeling with provable guarantees. In *International Conference on Machine Learning*, pages 280–288, 2013.

[9] Mohammad Taha Bahadori, Krzysztof Chalupka, Edward Choi, Robert Chen, Walter F. Stewart, and Jimeng Sun. Causal regularization. In *arXiv pre-print*, 2017.

[10] David J Bartholomew, Martin Knott, and Irini Moustaki. *Latent variable models and factor analysis: A unified approach*, volume 904. John Wiley & Sons, 2011.

[11] Justin Basilico and Yves Raimond. Déjà vu: The importance of time and causality in recommender systems. In *Proceedings of the Eleventh ACM Conference on Recommender Systems*, pages 342–342. ACM, 2017.

[12] Julien Bensmail, Marthe Bonamy, and Hervé Hocquard. Strong edge coloring sparse graphs. *Electronic Notes in Discrete Mathematics*, 49:773–778, 2015.

[13] David Berthelot, Thomas Schumm, and Luke Metz. Began: Boundary equilibrium generative adversarial networks. In *arXiv pre-print*, 2017.

[14] Michel Besserve, Naji Shajarisales, Bernhard Schölkopf, and Dominik Janzing. Group invariance principles for causal generative models. In *Proceedings of the 21st International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2017.

[15] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022, 2003.

[16] Ashish Bora, Ajil Jalal, Eric Price, and Alexandros G. Dimakis. Compressed sensing using generative models. In *ICML 2017*, 2017.

[17] Sofia Borboudakis, Giorgos andTriantafillou and Ioannis Tsamardinos. Tools and algorithms for causally interpreting directed edges in maximal ancestral graphs. In *Sixth European Workshop on Probabilistic Graphical Models*, 2012.

[18] Matthew Brand. An entropic estimator for structure discovery. In *Advances in Neural Information Processing Systems*, pages 723–729, 1999.

[19] Richard Caron and Tim Traynor. The zero set of a polynomial, 2005. [Online].

[20] Krzysztof Chalupka, Tobias Bischoff, Pietro Perona, and Frederick Eberhardt. Unsupervised discovery of el nino using causal feature learning on microlevel climate data. In *Proc. of UAI'16*, 2016.

[21] Yan Chen, Xi Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In *Proceedings of NIPS 2016*, Barcelona, Spain, December 2016.

[22] Zhitang Chen, Kun Zhang, Laiwan Chan, and Bernhard Schölkopf. Causal discovery via reproducing kernel hilbert space embeddings. *Neural Computation*, 26:1484–1517, 2014.

[23] David Maxwell Chickering. Optimal structure identification with greedy search. *Journal of Machine Learning Research*, 3:507–554, 2002.

[24] Tom Claassen and Tom Heskes. Causal discovery in multiple models from different experiments. In *Advances in Neural Information Processing Systems*, pages 415–423, 2010.

[25] C. Clopper and E. S. Pearson. The use of confidence or fiducial limits illustrated in the case of the binomial. *Biometrika*, 26:404–413, 1934.

[26] A. Philip Dawid. Beware of the dag! In *Proceedings of NIPS Workshop on Causality*, 2008.

[27] Jesús De Loera and Edward D. Kim. Combinatorics and geometry of transportation polytopes: An update. In *Discrete Geometry and Algebraic Combinatorics, volume 625 of Contemporary Mathematics*, pages 37–76, 2014.

[28] Dua Dheeru and Efi Karra Taniskidou. UCI machine learning repository, 2017.

[29] Chris Donahue, Akshay Balsubramani, Julian McAuley, and Zachary C. Lipton. Semantically decomposing the latent spaces of generative adversarial networks. In *Proc. of ICLR*, 2018.

[30] Jeff Donahue, Philipp Krähenbühl, and Trevor Darrell. Adversarial feature learning. In *ICLR*, 2017.

[31] Diego Delle Donne and Javier Marenco. Polyhedral studies of vertex coloring problems: The standard formulation. *Discrete Optimization*, 21:1–13, 2016.

[32] Vincent Dumoulin, Ishmael Belghazi, Ben Poole, Olivier Mastropietro, Alex Lamb, Martin Arjovsky, and Aaron Courville. Adversarially learned inference. In *ICLR*, 2017.

[33] Frederich Eberhardt, Clark Glymour, and Richard Scheines. On the number of experiments sufficient and in the worst case necessary to identify all causal relations among n variables. In *Proceedings of the 21st Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 178–184, 2005.

[34] Frederick Eberhardt. Causation and intervention. *(Ph.D. Thesis)*, 2007.

[35] Frederick Eberhardt. Almost optimal intervention sets for causal discovery. In *Proceedings of 24th Conference in Uncertainty in Artificial Intelligence (UAI)*, pages 161–168, 2008.

[36] Frederick Eberhardt and Richard Scheines. Interventions and causal inference. *Philosophy of Science*, 74(5):981–995, 2007.

[37] Jalal Etesami and Negar Kiyavash. Discovering influence structure. In *IEEE ISIT*, 2016.

[38] Thiago de Paulo Faleiros, Alneu de Andrade Lopes, et al. On the equivalence between algorithms for non-negative matrix factorization and latent dirichlet allocation. In *European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning, XXIV*. European Neural Network Society-ENNS, 2016.

[39] András Frank. Some polynomial algorithms for certain graphs and hypergraphs. In *Proc. of the Fifth British Combinatorial Conference, Congressus Numerantium XV*, 1975.

[40] Weihao Gao, Sreeram Kannan, Sewoong Oh, and Pramod Viswanath. Causal strength via shannon capacity: Axioms, estimators and applications. In *Proceedings of the 33 rd International Conference on Machine Learning*, 2016.

[41] Eric Gaussier and Cyril Goutte. Relation between plsa and nmf and implications. In *Proceedings of the 28th annual international ACM*

SIGIR conference on Research and development in information retrieval, pages 601–602. ACM, 2005.

[42] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Proceedings of NIPS 2014*, Montreal, CA, December 2014.

[43] Olivier Goudet, Diviyan Kalainathan, Philippe Caillou, David Lopez-Paz, Isabelle Guyon, Michele Sebag, Aris Tritas, and Paola Tubaro. Learning functional causal models with generative neural networks. In *arXiv pre-print*, 2017.

[44] Matthew Graham and Amos Storkey. Asymptotically exact inference in differentiable generative models. In Aarti Singh and Jerry Zhu, editors, *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *Proceedings of Machine Learning Research*, pages 499–508, Fort Lauderdale, FL, USA, 20–22 Apr 2017. PMLR.

[45] Clive W. Granger. Investigating causal relations by econometric models and cross-spectral methods. *Econometrica: Journal of the Econometric Society*, pages 424–438, 1969.

[46] Moritz Grosse-Wentrup, Dominik Janzing, Markus Siegel, and Bernhard Schölkopf. Identification of causal relations in neuroimaging data with latent confounders: An instrumental variable approach. *NeuroImage*, 125:825–833, 2016.

[47] Moritz Grosse-Wentrup, Dominik Janzing, Markus Siegel, and Bernhard Schölkopf. Identification of causal relations in neuroimaging data with latent confounders: An instrumental variable approach. *NeuroImage (Elsevier)*, 125:825–833, 2016.

[48] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron Courville. Improved training of wasserstein gans. In *Proc. of NIPS*, 2017.

[49] Alain Hauser and Peter Bühlmann. Characterization and greedy learning of interventional markov equivalence classes of directed acyclic graphs. *Journal of Machine Learning Research*, 13(1):2409–2464, 2012.

[50] Alain Hauser and Peter Bühlmann. Two optimal strategies for active learning of causal networks from interventional data. In *Proceedings of Sixth European Workshop on Probabilistic Graphical Models*, 2012.

[51] Alain Hauser and Peter Bühlmann. Two optimal strategies for active learning of causal models from interventional data. *International Journal of Approximate Reasoning*, 55(4):926–939, 2014.

[52] David Heckerman. A tutorial on learning with bayesian networks. *Tech. Rep. MSR–TR–95–06, Microsoft Research*, 1995.

[53] Thomas Hofmann. Probabilistic latent semantic analysis. In *Proceedings of the Fifteenth conference on Uncertainty in artificial intelligence*, pages 289–296. Morgan Kaufmann Publishers Inc., 1999.

[54] Patrik O Hoyer, Dominik Janzing, Joris Mooij, Jonas Peters, and Bernhard Schölkopf. Nonlinear causal discovery with additive noise models. In *Proceedings of NIPS 2008*, 2008.

[55] Huining Hu, Zhentao Li, and Adrian Vetta. Randomized experimental design for causal graph discovery. In *Proceedings of NIPS 2014*, Montreal, CA, December 2014.

[56] Antti Hyttinen, Frederick Eberhardt, and Patrik Hoyer. Experiment selection for causal discovery. *Journal of Machine Learning Research*, 14:3041–3071, 2013.

[57] Antti Hyttinen, Frederick Eberhardt, and Patrik O Hoyer. Causal discovery of linear cyclic models from multiple experimental data sets with overlapping variables. *Proceedings of the 28th Conference on Uncertainty in Artificial Intelligence (UAI)*, 2012.

[58] Antti Hyttinen, Patrik O Hoyer, Frederick Eberhardt, and Matti Jarvisalo. Discovering cyclic causal models with latent variables: A general sat-based procedure. *In Proceedings of the 29th Conference on Uncertainty in Artificial Intelligence (UAI)*, 2013.

[59] Guido W. Imbens and Donald B. Rubin. *Causal Inference for Statistics, Social, and Biomedical Sciences: An Introduction.* Cambridge University Press, 2015.

[60] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proc. of CVPRâĂŹ17*, 2017.

[61] Dominik Janzing, Joris Mooij, Kun Zhang, Jan Lemeire, Jakob Zscheischler, Povilas Daniusis, Bastian Steudel, and Bernhard Schölkopf. Information-geometric approach to inferring causal directions. *Artificial Intelligence*, 182-183:1–31, 2012.

[62] Dominik Janzing, Jonas Peters, Joris Mooij, and Bernhard Schölkopf. Identifying confounders using additive noise models. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, pages 249–257. AUAI Press, 2009.

[63] Dominik Janzing, Eleni Sgouritsa, Oliver Stegle, Jonas Peters, and Bernhard Schölkopf. Detecting low-complexity unobserved causes. *Proc. of UAI'11*, 2011.

[64] Theofanis Karaletsos. Adversarial message passing for graphical models. In *Proceedings of NIPS 2016*, Barcelona,Spain, December 2016.

[65] Gyula Katona. On separating systems of a finite set. *Journal of Combinatorial Theory*, 1(2):174–194, 1966.

[66] Gyula Katona. On separating systems of a finite set. *Journal of Combinatorial Theory*, 1(2):174–194, 1966.

[67] Murat Kocaoglu, Alexandros G. Dimakis, and Sriram Vishwanath. Learning causal graphs with constraints. In *NIPS 2016 Workshop: What If? Inference and Learning of Hypothetical and Counterfactual Interventions in Complex Systems*, 2016.

[68] Murat Kocaoglu, Alexandros G. Dimakis, and Sriram Vishwanath. Cost-optimal learning of causal graphs. In *ICML'17*, 2017.

[69] Murat Kocaoglu, Alexandros G. Dimakis, Sriram Vishwanath, and Babak Hassibi. Entropic causal inference. In *AAAI'17*, 2017.

[70] Murat Kocaoglu, Alexandros G. Dimakis, Sriram Vishwanath, and Babak Hassibi. Entropic causality and greedy minimum entropy coupling. In *ISIT'17*, 2017.

[71] Murat Kocaoglu*, Karthikeyan Shanmugam*, and Elias Bareinboim. Experimental design for learning causal graphs with latent variables. In *Proc. NIPS'17*, 2017.

[72] Murat Kocaoglu*, Karthikeyan Shanmugam*, Alex Dimakis, and Adam Klivans. Sparse polynomial learning and graph sketching. In *Proc. NIPS'14 (oral)*, 2014.

[73] Murat Kocaoglu*, Christopher Snyder*, Alexandros G. Dimakis, and Sriram Vishwanath. Causalgan: Learning causal implicit generative models with adversarial training. In *Proc. of ICLR*, 2017.

[74] Ioannis Kontoyiannis and Maria Skoularidou. Estimating the directed information and testing for causality. *IEEE Transactions on Information Theory*, 62:6053–6067, Aug. 2016.

[75] Mladen Kovacevic, Ivan Stanojevic, and Vojin Senk. On the hardness of entropy minimization and related problems. In *IEEE Information Theory Workshop*, 2012.

[76] Richard J Lipton and Robert Endre Tarjan. A separator theorem for planar graphs. *SIAM Journal on Applied Mathematics*, 36(2):177–189, 1979.

[77] Furui Liu and Laiwan Chan. Causal discovery on discrete data with extensions to mixture model. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 7(2):21, 2016.

[78] Ming-Yu Liu, Thomas Breuel, and Jan Kautz. Unsupervised image-to-image translation. In *Proc. of the 31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.*, 2017.

[79] Ming-Yu Liu and Tuzel Oncel. Coupled generative adversarial networks. In *Proceedings of NIPS 2016*, Barcelona,Spain, December 2016.

[80] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

[81] Po-Ling Loh and Peter Bühlmann. High-dimensional learning of linear causal networks via inverse covariance estimation. *Journal of Machine Learning Research*, 5:3065–3105, 2014.

[82] David Lopez-Paz, Krikamol Muandet, and Benjamin Recht. The randomized causation coefficient. *Journal of Machine Learning Research*, 16:2901–2907, Dec. 2015.

[83] David Lopez-Paz, Krikamol Muandet, Bernhard Schölkopf, and Ilya Tolstikhin. Towards a learning theory of cause-effect inference. In *Proceedings of ICML 2015*, 2015.

[84] David Lopez-Paz, Robert Nishihara, Soumith Chintala, Bernhard Schölkopf, and Léon Bottou. Discovering causal signals in images. In *Proceedings of CVPR 2017*, Honolulu, CA, July 2017.

[85] David Lopez-Paz and Maxime Oquab. Revisiting classifier two-sample tests. In *Proc. ICLR'17*, 2017.

[86] Sara Magliacane, Tom Claassen, and Joris M Mooij. Joint causal inference on observational and experimental datasets. *arXiv preprint arXiv:1611.10351*, 2016.

[87] Cai Mao-Cheng. On separating systems of graphs. *Discrete Mathematics*, 49:15–20, 1984.

[88] Dimitris Margaritis. Learning bayesian network model structure from data. Technical report, CARNEGIE-MELLON UNIV PITTSBURGH PA SCHOOL OF COMPUTER SCIENCE, 2003.

[89] MCI. Munich workshop on causal inference and information theory. https://www.lnt.ei.tum.de/events/munich-workshop-on-causal-inference-and-information-theory-2016/, 2016. Accessed: 2016 - 09 - 14.

[90] Christopher Meek. Causal inference and causal explanation with background knowledge. In *Proc. of the 11th International Conference on Uncertainty in Artificial Intelligence*, 1995.

[91] Christopher Meek. Strong completeness and faithfulness in bayesian networks. In *Proceedings of the Eleventh conference on Uncertainty in artificial intelligence*, pages 411–418. Morgan Kaufmann Publishers Inc., 1995.

[92] Christopher Meek. Strong completeness and faithfulness in bayesian networks. In *Proceedings of the eleventh international conference on uncertainty in artificial intelligence*, 1995.

[93] Stijn Meganck, Sam Maes, Philippe Leray, and Bernard Manderick. Learning semi-markovian causal models using experiments. In *Proceedings of The third European Workshop on Probabilistic Graphical Models , PGM 06*, 2006.

[94] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. In *arXiv pre-print*, 2014.

[95] Shakir Mohamed and Balaji Lakshminarayanan. Learning in implicit generative models. In *arXiv pre-print*, 2016.

[96] J. M. Mooij, D. Janzing, J. Zscheischler, and B. Schölkopf. Cause effect pairs repository, 2016. [Online].

[97] Joris M. Mooij, Oliver Stegle, Dominik Janzing, Kun Zhang, and Bernhard Schölkopf. Probabilistic latent variable models for distinguishing between cause and effect. In *Proceedings of NIPS 2010*, 2010.

[98] M. Joris Mooij, Jonas Peters, Dominik Janzing, Jakob Zscheischler, and Bernhard Schölkopf. Distinguishing cause from effect using observational data: methods and benchmarks. *Journal of Machine Learning Research*, 17(32):1–102, 2016.

[99] K. G. Murty and S. K. Kabadi. Some np-complete problems in quadratic and nonlinear programming. *Math. Programming*, 39:117–129, 1987.

[100] Christian Naesseth, Francisco Ruiz, Scott Linderman, and David Blei. Reparameterization Gradients through Acceptance-Rejection Sampling Algorithms. In Aarti Singh and Jerry Zhu, editors, *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *Proceedings of Machine Learning Research*, pages 489–498, Fort Lauderdale, FL, USA, 20–22 Apr 2017. PMLR.

[101] Augustus Odena, Christopher Olah, and Jonathon Shlens. Conditional image synthesis with auxiliary classifier gans. In *Proc. of the 34th International Conference on Machine Learning (ICML)*, 2017.

[102] Shmuel Onn and Ishay Weissman. Generating uniform random vectors over a simplex with implications to the volume of a certain polytope and to multivariate extremes. *Annals of Operations Research*, 189:331–342, 2011.

[103] Pekka Parviainen and Mikko Koivisto. Ancestor relations in the presence of unobserved variables. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 2011.

[104] J. Pearl, M. Glymour, and N.P. Jewell. *Causal Inference in Statistics: A Primer*. Wiley, 2016.

[105] Judea Pearl. *Causality: Models, Reasoning and Inference*. Cambridge University Press, 2009.

[106] Jonas Peters and Peter Bühlman. Identifiability of gaussian structural equation models with equal error variances. *Biometrika*, 101:219–228, 2014.

[107] Jonas Peters, Peter BÃijhlmann, and Nicolai Meinshausen. Causal inference using invariant prediction: identification and confidence intervals. *Statistical Methodology, Series B*, 78:947 – 1012, 2016.

[108] Jonas Peters, Dominik Janzing, and Bernhard Schölkopf. Causal inference on discrete data using additive noise models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33:2436–2450, 2011.

[109] Janet Piñero, Núria Queralt-Rosinach, Àlex Bravo, Jordi Deu-Pons, Anna Bauer-Mehren, Martin Baron, Ferran Sanz, and Laura I Furlong. Disgenet: a discovery platform for the dynamical exploration of human diseases and their genes. *Database*, 2015, 2015.

[110] Christopher Quinn, Negar Kiyavash, and Todd Coleman. Directed information graphs. *IEEE Transactions on Information Theory*, 61:6887–6909, Dec. 2015.

[111] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. In *arXiv pre-print*, 2015.

[112] Maxim Raginsky. Directed information and pearl's causal calculus. In *Proc. 49th Annual Allerton Conf. on Communication, Control and Computing*, 2011.

[113] Joseph D. Ramsey Ramsey, Stephen José Hanson, Catherine Hanson, Yaroslav O. Halchenko, Russell Poldrack, and Clark Glymour. Six problems for causal inference from fmri. *NeuroImage (Elsevier)*, 49:1545–1558, 2010.

[114] Mateo Rojas-Carulla, Marco Baroni, and David Lopez-Paz. Causal discovery using proxy variables. In *arXiv pre-print*, 2017.

[115] Donald Rubin. Estimating causal effects of treatments in randomized and nonrandomized studies. *Journal of Educational Psychology*, 66 (5):688–701, 1974.

[116] Federica Russo. *Causality and causal modelling in the social sciences*. Springer, 2010.

[117] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. In *NIPS'16*, 2016.

[118] Bernhard Schölkopf, David W. Hogg, Dun Wang, Daniel Foreman-Mackey, Dominik Janzing, Carl-Johann Simon-Gabriel, and Jonas Peters. Removing systematic errors for exoplanet search via latent causes. In *in Proc. of the 32nd International Conference on Machine Learning*, 2015.

[119] Eleni Sgouritsa, Dominik Janzing, Jonas Peters, and Bernhard Schölkopf. Identifying finite mixtures of nonparametric product distributions and causal inference of confounders. *Proceedings of the 29th Conference on Uncertainty in Artificial Intelligence (UAI 2013)*, 2013.

[120] Naji Shajarisales, Dominik Janzing, Bernhard Schölkopf, and Michel Besserve. Telling cause from effect in deterministic linear dynamical

systems. In *Proceedings of the 32 nd International Conference on Machine Learning*, 2015.

[121] Karthikeyan Shanmugam*, Murat Kocaoglu*, Alex Dimakis, and Sujay Sanghavi. Sparse quadratic logistic regression in sub-quadratic time. In *arXiv preprint*, 2017.

[122] Karthikeyan Shanmugam, Murat Kocaoglu, Alex Dimakis, and Sriram Vishwanath. Learning causal graphs with small interventions. In *NIPS 2015*, 2015.

[123] Madhusudana Shashanka, Bhiksha Raj, and Paris Smaragdis. Sparse overcomplete latent variable decomposition of counts data. In *Advances in neural information processing systems*, pages 1313–1320, 2008.

[124] S Shimizu, P. O Hoyer, A Hyvarinen, and A. J Kerminen. A linear non-gaussian acyclic model for causal discovery. *Journal of Machine Learning Research*, 7:2003–âĂŞ2030, 2006.

[125] Ricardo Silva, Richard Scheines, Clark Glymour, and Peter Spirtes. Learning the structure of linear latent variable models. *Journal of Machine Learning Research*, 7:191–246, 2006.

[126] Peter Spirtes, Clark Glymour, and Richard Scheines. *Causation, Prediction, and Search*. A Bradford Book, 2001.

[127] Kumar Sricharan. Personal communication., 2017.

[128] Kumar Sricharan, Raja Bala, Matthew Shreve, Hui Ding, Kumar Saketh, and Jin Sun. Semi-supervised conditional gans. In *arXiv pre-print*, 2017.

[129] Mervyn Susser. Glossary: causality in public health science. *Journal of Epidemiology & Community Health*, 55(6):376–378, 2001.

[130] Jin Tian and Judea Pearl. Causal discovery from changes. In *Proceedings of UAI2013*, 2013.

[131] Luan Tran, Xi Yin, and Xiaoming Liu. Disentangled representation learning gan for pose-invariant face recognition. 2017.

[132] Sofia Triantafillou and Ioannis Tsamardinos. Constraint-based causal discovery from multiple interventions over overlapping variable sets. *Journal of Machine Learning Research*, 16:2147–2205, 2015.

[133] Sofia Triantafillou, Ioannis Tsamardinos, and Ioannis Tollis. Learning causal structure from overlapping variable sets. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pages 860–867, 2010.

[134] Caroline Uhler, Garvesh Raskutti, Peter BÃijhlmann, and Bin Yu. Geometry of the faithfulness assumption in causal inference. *Annals of Statistics*, 41:436–463, 2013.

[135] Thomas Verma and Judea Pearl. An algorithm for deciding if a set of observed independencies has a causal explanation. In *Proceedings of the*

Eighth international conference on uncertainty in artificial intelligence, 1992.

[136] Carl Vondrick, Hamed Pirsiavash, and Antonio Torralba. Generating videos with scene dynamics. In *Proceedings of NIPS 2016*, Barcelona, Spain, December 2016.

[137] Simon Webb. Central slices of the regular simplex. *Geometriae Dedicata*, 61(1):19–28, 1996.

[138] Ingo Wegener. On separating systems whose elements are sets of at most k elements. *Discrete Mathematics*, 28(2):219–222, 1979.

[139] Mirjam Weilenmann and Roger Colbeck. Analysing causal structures with entropy. *Proc. R. Soc. A*, 473(2207):20170483, 2017.

[140] Lijun Wu, Yingce Xia, Li Zhao, Fei Tian, Tao Qin, Jianhuang Lai, and Tie-Yan Liu. Adversarial neural machine translation. In *arXiv pre-print*, 2017.

[141] Mihalis Yannakakis and Fanica Gavril. The maximum k-colorable subgraph problem for chordal graphs. *Information Processing Letters*, 24:133–137, 1987.

[142] Jiji Zhang. Causal reasoning with ancestral graphs. *J. Mach. Learn. Res.*, 9:1437–1474, June 2008.

[143] Jiji Zhang. On the completeness of orientation rules for causal discovery in the presence of latent confounders and selection bias. *Artificial Intelligence*, 172(16):1873–1896, 2008.

[144] Junbo Zhao, Michael Mathieu, and Yann LeCun. Energy-based generative adversarial networks. In *ICLR*, 2017.

[145] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *IEEE International Conference on Computer Vision (ICCV)*, 2017.

[146] Brian D. Ziebart, J. Andrew Bagnell, and Anind K. Dey. The principle of maximum causal entropy for estimating interacting processes. *IEEE Transactions on Information Theory*, 59:1966 – 1980, 2013.

Vita

Murat Kocaoglu received his Bachelor of Science degree in Electrical - Electronics Engineering with a minor degree in Physics from Middle East Technical University, and Master of Science degree from Koc University, Turkey. He received his PhD degree at The University of Texas at Austin in the Wireless Networking and Communications Group (WNCG), under the supervision of Prof. Alex Dimakis and Prof. Sriram Vishwanath. His current research interests are on causal inference, generative models, learning theory and information theory.

E-mail: mkocaoglu@utexas.edu

This dissertation was typeset with LaTeX[†] by the author.

[†]LaTeX is a document preparation system developed by Leslie Lamport as a special version of Donald Knuth's TeX Program.